uc3m | Universidad Carlos III de Madrid

University Degree in Computer Science and Engineering
and Business Administration
2018-2019

Bachelor Thesis

Photo Enhancement On Mobile Devices Using Deep Neural Networks

---

Guillermo Fragío Sánchez

Tutor:

Miguel Ángel Patricio Guisado

Colmenarejo, 2019

ABSTRACT

In recent years, the return of the usage of Artificial Neural Networks has lead to the greatest improvements in the field of Artificial Intelligence, due to the huge diversity of different applications that deep learning models has in a large variety of research fields, and also the evolution of information processing systems capacity. This thesis aims to study which deep neural networks models are most suitable for photo enhancement, to generate images with certain desired characteristics.

Model selection has been done by comparing the both supervised, Convolutional Neural Networks, and unsupervised models, Generative Adversarial Networks. It has been demonstrated that Generative Adversarial Networks have great potential by showing results that compete with the state of the art. The chosen model is a Generative Adversarial model which outperforms the rest in terms of a combination of enhancement quality and time taken in the process. Moreover, since the model is compatible with mobile devices it has been integrated and evaluated in a BQ smartphone, to proof its viability on mobile devices.

Keywords: Computer Vision, Photo enhancement, Deep Neural Networks, Generative Adversarial Networks, Smartphone, Mobile Application.

ACKNOWLEDGEMENTS

Completing my Bachelor´s degree has been a long and exciting journey. Now that I have reached the final stage, I would like to take some time to thank my family for giving me the opportunity to study this degree, and for their unceasing encourgament and support through the last years. To my friends for giving me their support to always keep going forward.

I would also like to express my gratitude to Miguel Ángel Patricio Guisado for its guidance throughout the project, and to all the professors who have been involved over the last years.

Finally, to my classmates and everyone who, in one way or another, have played a part to get to this point.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

MLP: Multilayer Perceptron

CNN: Convolutional Neural Networks

GAN: Generative Adversarial Networks

PSNR: Peak Signal to Noise Ratio

SSIM: Structural Similarity Index

RGB: Red Green Blue channels.

1. INTRODUCTION

1.1 Motivation

Taking pictures has been and it remains as a way to save our memories. However, the process for taking photographs has not been as easy as it is nowadays with the revolutionary evolution of digital cameras and smartphones. In 19th century, first photogrpah was taken by the french inventor Joseph Nicéphore Niépce using bitumen in the so called *camera obscur*a [**1**]. The process consisted in a mixture of chemistry and light exposure which was far from being automatic and more important, an instant process. Thus, qualified people were requiered to capture an image and it used to take some time to visualize the final result. The evolution from traditional cameras to digital ones, and after to campact cameras integrated on smartphones, has supossed a change in the field of photography. At present, people with low photography skills which have access to a mobile device, are able to take high quality images in just "one click".

Once an image is taken, it may need to be post-processed in order to get a new image with specific characteristics. One of these post-processing processes is **image enhancement.** In image processing, image enhancement can be addressed as a translation image process which aims to improve the visual appearance of an image to generate an enhanced one for a specific application [**2**]. The enhancement tries to highlight image features such as contrast, brightness, colour, etc., or to remove others as inherent noise or blur. In fields such as medicine or astronomy, this process is commonly used as a preprocessing method to make image comprehesion more suitable for humans or machines. There are several traditional techniques which are widely applied to resolve this issue as histogram equalization, contrast stretching, fuzzy logic [**2**] , [**3**] and also more innovative approchaes which use deep learning algorithms like Convolutional Neural Networks, which in the past years have shown a better performance in many image processsing and computer vision applications [**4**] .

Nevertheless, image enhancement can also be applied to just obtain a more pleasant image as profesional photography editors do i.e., **photo enhancement**. Good results in this task requieres the combination of three key elements: i) knowledge on how to retouch images, ii) skills to use high level software editing tools (Adobe Lightroom, Adobe Photoshop CC…) and more importantly, iii) time. Though editing tools have evolved and now provide automatic photo enhancement, in some cases, users have to pay a license to use them. So this add a fourth element to the equation, making the process a bit more tedious.

But, what if an artifical intelligence model could learn to reproduce this process automatically, combining profesional knowledge and an efficient use of time?. In the field of image processing, recent research which uses deep learning techniques yield very good performance in this task. This methods not only show (good) quality results but also to be efficient in terms of time. So as it happened in the past taking photographs, using deep learning techniques in photo enhancement make the process automatic and also accesible to everyone in a few seconds.

Moreover, the increasing use of social networks have produced a massive increase of people who upload images and personal photographies to the Interent. As a result, people are more aware about the content and visual appeareance of their photographs, so they try to retouch them using filters and other editing tools. Therefore, automatic photo enhancement is an issue which might be useful for anyone who has a Facebook or Instagram account.

1.2 Objectives

This Bachelor's Thesis **main objective** is to **evaluate**, **choose and apply a deep neural network model in photo enhancement task** in order to transform an input image into an enchanced one with specific characteristics. The chosen model has to perform efficiently a combination of enhancement quality and run time. The aim of the enhancement is not to have an output image which is better than the original in terms of noise, signal quality or resolution, but to generate an enhanced image with characteristics which are, in general, more suitable for users.

Thanks to the cooperation between University Carlos III de Madrid and the spanish cellular company BQ, we made a research to apply neural networks in photo enhancement. Cellular companies like BQ are interested in integrate Artificial Intelligence models in their smartphones due to the evolution of their computational capacity. Therefore, as the **second objective**, the chosen model is adapted to **reproduce the process in a mobile device** to check its viability and evaluate its performance on mobile devices.

However, it has to be clear that the aim of this study is not focused in the development of a new software product, but in the investigantion and evaluation of neural network architectures which try to solve photo enhancement problem.

1.3 Methodology

This thesis follows to different methodologies. The first part consists in a introduction of why neural networks are a reasonable approach to photo enhancement task, and a literary review of previous work which use deep learning models as a solution. The information and papers related have been collected from different sources as Google Scholar, IEEE Xplore, arXiv, among others.

The other part of the thesis is an experimental test where different models are evaluated based on previous established metrics and an analysis of the obtained results in order to choose the best performance.

1.4 Socio-economic environment

Since photogrpahs are part of our dalily life, photo enhancement task may have an impact in a social and economic way. First, companies, specially cellular producers, might be intereseted in the development of a photo enhancer to introduce it as a new feature in their smartphones, and take advantage with respect other competitors. This add extra value to their product so it is more enjoyable and useful for the photographs taken with them. As a result, this may lead to an sales increase and, therefore, to higher profits.

From a social point of view, the usage of social networks where there are constantly uploading new photographs, people is interested in sharing their experiences in the most enjoyable way as possible. In particular, the new figure called “influencer” has taken an important relevance in the world of internet. These people make from their daily life a job with which, thanks to sponsors, they earn money while sharing their personal life to the public in social networks. So it is important for them to preserve a good looking life to show to their public. Photo enhancer may be useful for them to be more efficient while retouching photographs, specially for people which might not be interested in photography.

1.5 Regulatory framework

The development of this thesis involves using images where people might appear. According to the actual LOPD [**5**](Organic Law on Data Protection) legislation in Spain, which is where this research is taking place, the images used are taken from public datasets which have been made by the original authors os the work where it has been extracted or they are open to all who wants to use them, as is the case of MIT-5K dataset.

The implementation of the code is extracted from Github. Github counts with license to make the repository truly open source [**6**].The code used to replicate the results of the works which are presented in this document are open source, and have a MIT license to allow other developers to modify, distribute or make a private use of the code.

1.6 Document structure

The document's structure is divided in 4 main sections, so there is a logical order of how objectives are acheived.

First section consists in the state of the art, which encompasses the main reason why neural networks are the chosen technique as an approach to the task. There is also a revision of the evaluation metrics which are used in the evaluation of the selected models. And finally, previous approaches of photo enhancement task using both Convolutional Neural Networks and Generative Adversarial Networks.

In the second section, prior to evaluation, pre-trained model concept is introduced to clarify how the models are obtained before being evaluated. After, an experimental environment is set to evaluate and select the best models, from the ones presented in the state of the art, using the evaluation metrics presented in the previous section.

The third section is focused on the mobile application development, which uses the selected models in the previous section. An evaluation of the results is also done to conclude which is the most suitable model.

Finally, the forth section closes this thesis with the final conclusions, checking whether or not the objectives have been successfullly achieved, and the future lines of work.

2. STATE OF THE ART

2.1 Introduction

In recent years, (a part/a portion) of technology evolution has been marked by the emergence of terms as Machine Learning, Data Mining, Computer Vision or Digital Image Processing. All of them are included in Artificial Intelligence (AI) world to improving our daily life in many ways: healthcare, security and surveillance, business, speech recognition [**7**]. Computer Vision and Digital Image Processing are the most known in images field. However, the boundaries of these two concepts are often overlaped due to nature of the sort of problems both try to solve.

On one hand, the aim of Computer Vision (CV) is well described by a quote of Professor Fei-Fei Li, Co-director of Standford's Human-Centered AI Institute [**8**,**9**]: "If we want machines to think, we need to teach them to see". The main goal of CV is to develop models to extract, analyze and understand the content of an image, i.e. emulate human vision when seeing an image. Therefore, the source or input in CV is always an image, but the output can be of different nature: image, image classification, image recognition, shape, size, colors…

On the other hand, Digital Image Processing (DIP) as authors in [**10**] define, is a technique to process digital images by digital computers. This process normally aims to enhance or retouch specific image features so it is more suitable for human comprehesion. In this way, the source or input in Digital Image Processing is an image and the output is a processed image.

Therefore, a boundary can be established depending on the nature of the output of each teachnique. In [**10**] suggest that this is an "artificial boundary", but they agree most of DIP tasks have an image as an output.

In our case of study, photo enhancement using deep neural networks would be in between of both fields, because both input and output are images but the proposed network architecture would have to learn the inherent distribution of images in order to create a generic model.

TABLE I

COMPUTER VISION AND IMAGE PROCESSING INPUT/OUTPUT COMPARISON

Technique	**Input**	**Output**
Computer Vision	Image	Processed image, or image analysis
Digital Image Processing	Image	Processed Image

2.2 Photo Enhancement vs Image Enhancement: Traditional techniques

Images can be defined as a 2-dimensional intensity function *f(x,y)* where *x, y* represent spatial coordinates, and *f* is a function representing the the gray level intensity of each pair of coordinates *x,y* based on its amplitude. A digital image, is a 2-D array representation of discrete samples *f[x,y]*, where each sample represent the corresponding quantized amplitude value. Therefore, each sample of the 2-D array can be thought as a pixel of the image. Pixel range in a gray scale image goes from 0 (black) to 255 (white). [**11**] [**12**]

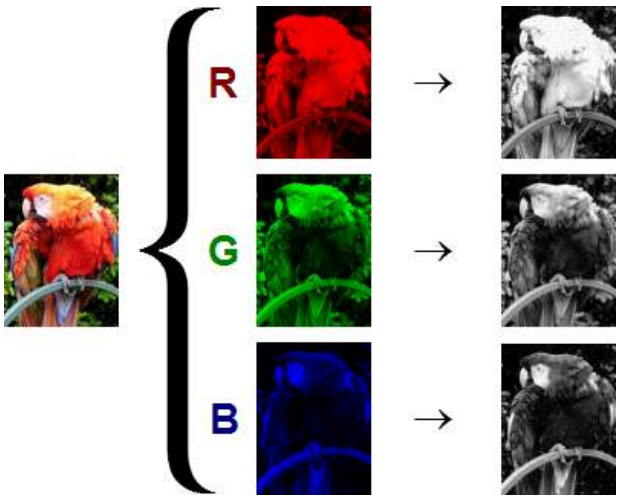

Fig 1. RGB channels separation. [66]

A coloured image, thus, is a combination of a set of three 2-D pixels array which are interpreted in three different channels: Red, Green and Blue (RGB).

Image enhancement techniques seek to process an image so that output image can be better analyzed by humans or machines. It accentuates certain features of the original image (edges, boundaries, contrast, histogram…) as required, however, it does not change image content data, but increases the dynamic range[1] of the chosen features. There exist a wide range of conventional techniques, which are used depending on the field of study. Contrast enhancement and spatial filtering techniques have therir applications in X-ray images, astrophotography and also surveillance, among others. [**3**] [**11**] [**2**]

[1] Ratio between the maximum and minimum values measurable light intensities.

Fig 2. Histogram equalization on X-Ray image. Left to right: input image / histogram equalization / adaptive histogram equalization. **[**68**]**

These techniques are useful as a preprocessing method which needs specific enhancement for an specific kind of images. For our purposes, these techniques do not solve photo enhancement issue.

Photo enhancement techniques, on the other hand, seek to edit an image to fix or to improve characteristics like exposure, brightness, textures… . Professional photographers use software tools like Adobe Photoshop CC or Adobe Lightroom due to they have plenty of options to edit photos manually. This tools require previous knowledge and skills in image editing. It also takes time to apply them and depending on photograph it takes more or less long to generate the desired enhanced photograph.

Fig 3. Lightroom manual editting. Left to right: original image / enhanced image. **[**69**]**

These photo enhancement techniques do not employ learning processes, and recent learning-based methods showed significant advantages over this traditional ones, thanks to the combination of both high capacity to analayse and extract information, and their possibilities to develop generic models.

Adobe Lightroom launched an edit button which uses artificial intelligence to better adjust photograph's parameters automatically [**13**]. This can be used by people with no experience and skills in editting, but this option is only available in Adobe Lightroom CC's premium.

There are also mobile applications as the one developed by Google, which recently rised a new functionality in Google Photos app to enhance photos automatically using artificial intelligence. It was presented at the Google I/O 2018 [**14**] and it includes automatic brightness adjustment and image colorization.

Also, Apple Photo Enhancer is an option included in Photos app in Apple mobile devices, which makes an automatic image enhancement using histogram values, face recognition and metadata properties to then apply an specific filter. However, they do not mention specifically if the process is guided by Artificial Intelligence techniques, but it can be assumed that a company as Apple has always best technology in their devices, and AI has shown great performance in image tasks.

2.3 Why Neural Networks? From Single Perceptron to Generative Adversarial Networks

2.3.1 Single and Multilayer Perceptron (MLP)

Artificial Neural Networks models have its origns in the past century. Since the appariton of the first logical model of neurons in 1943 [**15**] the evolution of Neural Networks have suffered ups and downs. They have not been as popular as they are nowadays.

The idea is based on human brain and its nervous system. In 1958, F.Rosenblat raised a model that imitates the behaviour of a biological neuron, the Perceptron [**16**] .The most basic unit in the model is called neuron, and its composition is inspired on the biological composition of a neuron.

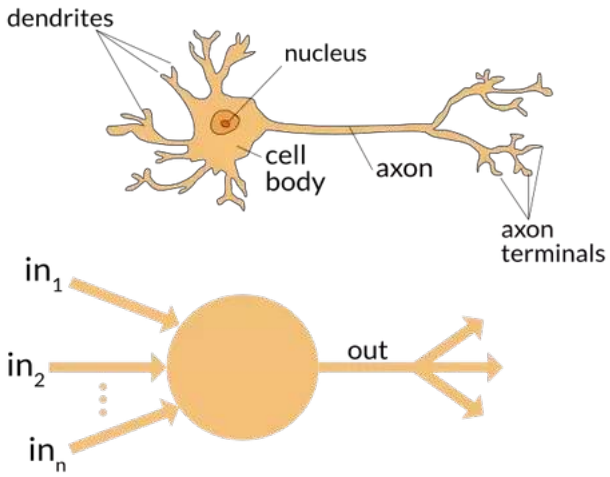

Fig 4. Biological neuron and Perceptron comparison. Top to bottom: Biological neuron / Perceptron.

In a synapse process an input signal is received in the dendrites. Once it has received enough signals (activation), they are passed through the axon and the outgoing signal is again connected to a set of dendrites of another neuron. Perceptron model mimic this process using a mathematical model, where the **input data** are binary elements, each of them multiplied by a number called **weight**. Then they are all summed, and a constant term called **bias** is added. The resul is finally thresholded to generate a binary output, 0 or 1. The activation of the neurons are modeled using an **activation function**, in this case thresholding, which measures the intensity of the stimulus of a neuron given certain values of its weights. The adjustment of weight values are then responsible of the activation of the neurons, and therefore of the learning process.

The learning algorithm was approached in a simple way. Given the real input and its corresponding output data, best called **training set**, and set of random weights:

1. Pick one training sample.
2. Compute Perceptron's output.
3. Compare Perceptron's output with real output value.
4. If output is not correct apply a method to modify weight values. For example:
    a. If real output is 0 and Perceptron's output is 1, decrease the weights.
    b. If real output is 1 and Perceptron's output is 0, increase the weights.
5. Repeat the process for each sample in the training set.

F.Rosenblat showed that this model can be used to classify images given real input-output pair, i.e. given the set of pixels of the image as the input data and the real output of what image represents. Although, Perceptron model itself was too weak to solve more than a binary classification problem.

Therefore, Perceptron was limited to distinguish only between 2 different classes. However, problems with more than two outputs, such as handwriting recognition, can be learned by multiple perceptrons. So the idea of a Neural Networks it's just a sequence of perceptrons (neurons) organized in so-called **layers**.

Multilayer Perceptron, or feedforward networks, architecture consists in: one input layer, one or more hidden layers, and one output layer. In this structure, each layer is composed by a set of neurons. Each of them is connected with all neurons in the next layer, and that's why they are called *fully connected layers*. Hidden layers are able to extract features from input data and allow other layers to operate with them, rather than using raw data. It also solves the non-linear classification problems of a single Perceptron. The problem is that previous learning algorithm does not fit with new

multilayer model, because the way weights are adjusted does not fit in a model with more than one layer. As a result, other training algorithms (learning algorithms) such as **backpropagation**, were raised to solve the propagation of the error in every layer and so weight adjustment is done to minimize the output error.

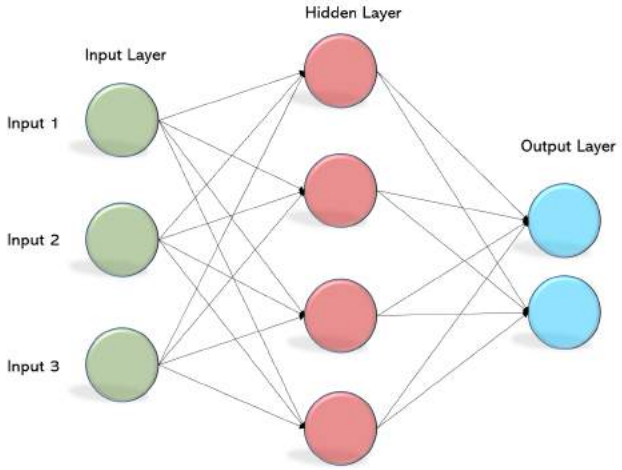

Fig 5. Multilayer Perceptron architecture overview. [70]

Until mid 1980's, researchers were intereseted in these revolutionary methods, and great advances were done [**17**] . Despite this, the bad performance of models, its poor scalability and the problems to train neura [**18**]l networks in a reasonable time, made researchers leave neural research behind.

During last deacade, the development of new powerful hardware and the high computational capacity of processors, make possible new research in deep learning techniques and new applications in Artificial Intelligence. Some of those possible applications were known before but further limitations did not allow to turn them into reality.

In the field of image processing and computer vision, Multilayer Perceptrons are not powerful enough when images are in a real world resolution. Therefore, other approaches such as Convolutional Neural Netoworks, and more recent Generative Adversarial Networks, have been successfully adopted as a better solution when dealing with images.

2.3.2 Training process

Learning of neural network happens during training process. The procedure to achieve it is called optimization algorithms or optimizers. These algorithms try to minimize a function called **loss function** which measures the performance of the network at each step of training. Loss function is normally composed by an **error term**, which

evaluates for each training sample the output of the network and the real output (correct output), and a **regularization term** which prevents the network of learning a very specific distribution of the data and failing when trying to generalise (overfitting). Optimizers, therefore, calculate at each training step which is the best weight adjustment to minimize the error term.

Training samples to feed the neural network are extracted as part of a dataset. A dataset is a collection of data from a specific topic used to train neural networks . It is divided into training set, to train the network, and test set, to test the performance of the network after training. Choosing a good training set is crucial since it relies on it how accurate the model is going to be. The more you feed a model with training samples, the better performance it has. However, training set cannot be very specific because it can lead to an overfitting issue in which model learns only some specific patterns, and afterwards, when tested with new unseen data, it fails trying to generalise. Therefore, choosing a consistent training set according to the target issue is also an important task to take into account, and in most of the cases, is as important as the architecture choice.

Training stops, or **convergence**, when the weights are correctly adjusted, i.e. when the error is minimized. Due to loss functions are complex sometimes error reaches a local minimum, and training stops before the error reaches the optimum minimization. The choice of which optimization algorithm to use depends on the issue one is trying to solve. [**18**] shows a compartive study of 5 different algorithms in terms of speed memory requirements and precision. For image tasks, Gradient Descent algorithms are the more suitable, because even if they perform more slow than others, memory use is minimized, which is very important when working with images.

2.3.3 Convolutional Neural Networks (CNN)

Neural Networks based on Multilayer Perceptrons architecture have limitations in image classification problems. The main and most important thing when dealing with images is being able to extract their features so they can be analysed and then use them in applications as image recognition, image classification, image processing, etc. Feature extraction using Multilayer Perceptron or deep traditional neural networks is not efficient because of the fully connectivity of hidden layers. To ilustrate this, an example is presented.

Imagine one have a grayscale image of 28x28 pixels, which represent 28 pixels height and 28 pixels width. And each of these pixels are the input data in a neural network, so in this case, there are 784 neurons in the first hidden layer. As it has seen before, each neuron is connected with all other neurons in the next hidden layer, and the next ones with the next hidden layer, and so on. This means that if neural network has n

$$Parameters = \prod_{i=0}^{n} Neurons_i \quad (1)$$

hidden layers, and each layer has $Neurons_i$ neurons, parameters grows to large as (1) shows. The problem is not just parameters high dimension but the inefficient way to use them, since neurons do not share them with each other, so information may be duplicated. In our example, managing 784 parameters in the first hidden layer is still doable, even with a bit higer dimension images as [**19**] demonstrate, results are acceptable. Nevertheless, images are too small, and in a real life situation images have 3 channels (RGB) and larger dimensions, since normal cameras image resolution is 640 x 480 pixels. Due to Multilayer Perceptrons were not able to handle with these images, another approach rised, Convolutional Neural Networks.

There are two main reasons why Convolutional Neural Networks perform better than Multilayer Perceptrons:

1. The feature extraction does not use a fully connected architecture, but sparse connectivity.
2. The use of filters in feature extraction allows CNN to extract specific patterns on the image with the advantage of sharing parameters, i.e. not getting redundant information.

CNN architecture explains how these two advantages are obtained with respect to MLP. The architecture is comprised of 3 types of layers: Convolution, Pooling and Fully Connected.

1) Convolution layer: it's composed by a set of kernels, also called filters. Each filter extracts an specific feature from the image (edges, colour, textures…), so the number of filters that are used reflects the number of features one wants to extract. The size of each filter is given depending on the architecture, but normally it has square dimensions (H = height, W = width, square dimensions: H x W, H = W) e.g. 3 x 3, 5 x 5. The layer's name comes from convolution process, in which a filter act as a window which is passed through the image to produce a, what is called, **feature map** or activation map. The feature map generated by each filter, represents another image which shows where the feature appears in the original image. Each element in the feature map represents a convolution of the original image. As Figure 6 shows, a *3 x 3* filter is passed taking a step of 1 pixel for each convolution. This step is called **Stride** and may vary between different architectures. Convolution changes image dimensions, the larger stride and filter size, the more spatial information is lost. However, it is possible to keep the original dimensions of the image after convolution by adding **zero-padding** to it before the convolution. Zero-padding refers to the set of pixels, with 0 value, which are introduced around the input image in order to get a desired dimension in the output image.

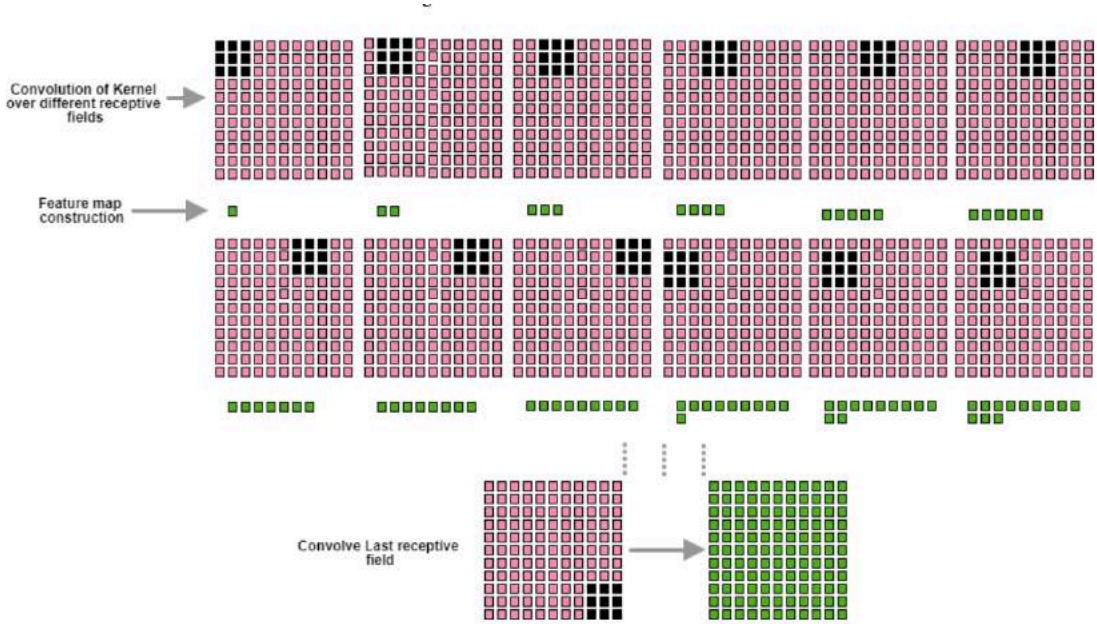

Fig 6. Convolution process in Convolutional Neural Netoworks. [71]

2) Pooling layer: the aim of pooling process is to reduce the dimension of the image and is normally done after one or more convolutions. The procedure is very similiar to convolution one, there is a chosen window and stride which passes over the image. There are different operations one can apply here, such as, max-pooling, average pooling, etc. For example, if one applies max-pooling, each element of the resultant feature map it's composed by the maximum value of the region of the window size. Since pooling reduces the dimension of the image, it also reduces the number of parameters, while reatining important information.

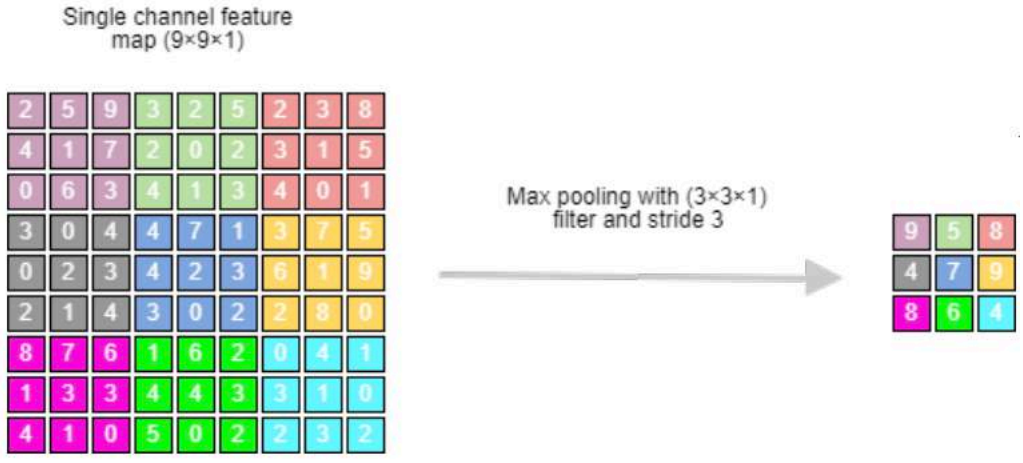

Fig 7. Pooling process in convolutional Neural Networks. [70]

3) Fully Connected layer: after extracting image features the image can be classified using a sequence of fully connected layers. This subnetwork architecture works as a Multilayer Perceptron, where output layer has as many outputs as elements to classify.

The number of layers of each type and the activation function, change within different architectures. In general, architectures are composed by a set of convolution layers and pooling layers in the middle, and ReLu activation function.

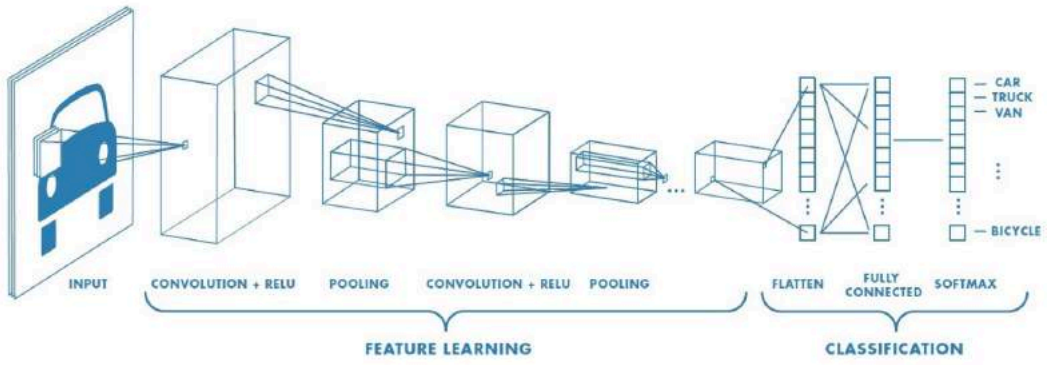

Fig 8. Convolutional Neural Network architecture overview. [20]

The difference between MLP and CNN is that in MLP architectures, the filter (which represents the weight values) and the corresponding elements generated in the feature map (neurons) are all connected with each other. Unlike, CNN applies sparse connectivity, which consists in just connect an area of the image, called receptive field, to only a part of the hidden layers. In this way, each neuron is "specialized" in a region of the image, and thus, perform a more precise feature extraction. Moreover, weights values of the same feature map are shared by the neurons, which allows parameter reduction.

CNN has been showing impressive results in image tasks. ImageNet Large Scale Visual Recognition Challenge [**21**] is one of the most important competition in image classification, in which participants have to classify images from 1000 different categories. Since 2012, competitors have improved previous results each year using CNN architectures. AlexNet was the first CNN architecture to win the challenge in 2012 [**22**] with a 16% error rate, and SENet [**23**] last one in 2017 with a 2,3% error rate.

2.3.4 Generative Adversarial Networks (GAN)

Multilayer Perceptron and Convolutional Neural Networks are tipically used in predictive modelling problems, which are trained by showing input data samples (X), making the model's prediction (Y'), and correcting the model with the real output (Y), i.e. "learn a mapping from inputs (X) and outputs (Y), given a labeled set of input-output pairs" [**24**]. This learning process is also known as **supervised learning**. Classification problems as image classification are predictive models and they are also referred as discriminative models, because aim to distinguish or discriminate between a set of given classes.

Generative Adversarial Networks, on the other hand, are a deep learning approach of generative modelling problems. Unlike predictive models, the generative model only uses input data samples (*X*), no outputs are given, and no corrections are made. These models are defined as **unsupervised learning** models: "*The second main type of machine learning is the descriptive or unsupervised learning approach. Here we are only given inputs, and the goal is to find "interesting patterns" in the data."* [**24**].Gaussian Mixture Models are a good example of a generative model algorithms used in data clustering [**25**].

Generative Adversarial Networks were first presented by Ian Goodfellow, et al. [**26**] in 2014 as:

"New framework for estimating generative models via an adversarial process, in which we simultaneously train two models: a generative model G that captures the data distribution, and a discriminative model D that estimates the probability that a sample came from the training data rather than G.".

They proposed a model where the Generator and Discriminator models are approached as Multilayer Perceptron's which compete with each other. In general, the Generator (*G*) learns the distribution of the data in order to generate new plausible samples which follows the input data distribution. And by his side, the Discriminator (*D*) is used as a classifier which aims to decide whether a sample is real (original distribution) or fake (generated by Generator). In other words, generator try to fool the discriminator generating samples which are intented to come from original data while discriminator aim to catch the new generated fake data from the generator.

In 2016, Ian Goodfellow made a report from a GAN tutorial done by Neural Information Processing Systems (NIPS) authors [**27**] which summarizes a set of the most important questions about this networks, and presents the GAN framework divided in generator and discriminator model.

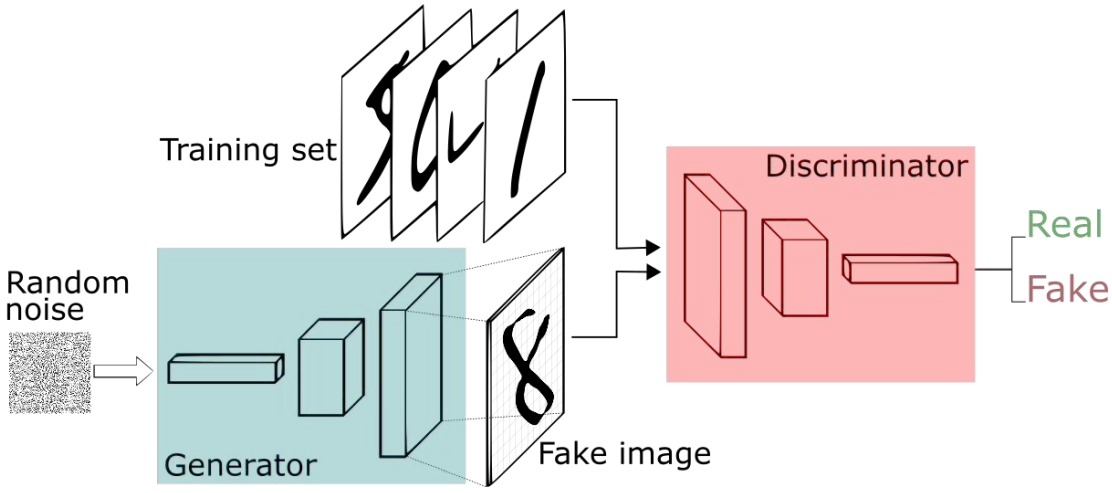

Fig 9. Generative Adversarial Network overview. [28]

Here, Ian remarks that GANs are a structured probabilistic model where the discriminator works with observed variables *x* and the generator with latent variables *z*. Latent variables are relevant variables or concepts which are hard or not directly observable. For example, an input vector to the generator model can be a random vector from a Gaussian distribution which after generates a sample in the problem domain.

- **Discriminator model**
  Discriminator is comprised of a function D, which takes *x* samples and learns to classify them between real or generated samples. The output *D(x)* is a numerical value which represents the probability that a sample *x* come from original data.

- **Generator model**
  Generator is comprised of a function G which learns the distribution of the observed domain by taking a noisy input vector of variables *z*. The output *G(z)* is a sample *x* which aims to be as most indistinguishable from original data as possible.

Functions *D* and *G* are tipically deep multilayer perceptrons models, which are trained simultaneously. The training process is a "max-min game" where discriminator tries to minimize *D(G(z))* while generator wants it to be maximized. Discriminator is fed with minibatches of *x* samples and generator with minibatches of noisy samples *z*. At first, in [**26**] they proposed updating discriminator's error at each step of training and the generator's error after *k* steps using backpropagation learning algorithm. Later, according to authors in best practices update the error in a strict simultaneous way,

one step for each model, using Stohastic Gradient Descent as learning algorithm. Ideally, training convergence is reached when discriminator classifies generated samples to be equally likely real or fake, i.e. $D(G(z)) = 0.5$. However, it does not have to reach that point to yield good results.

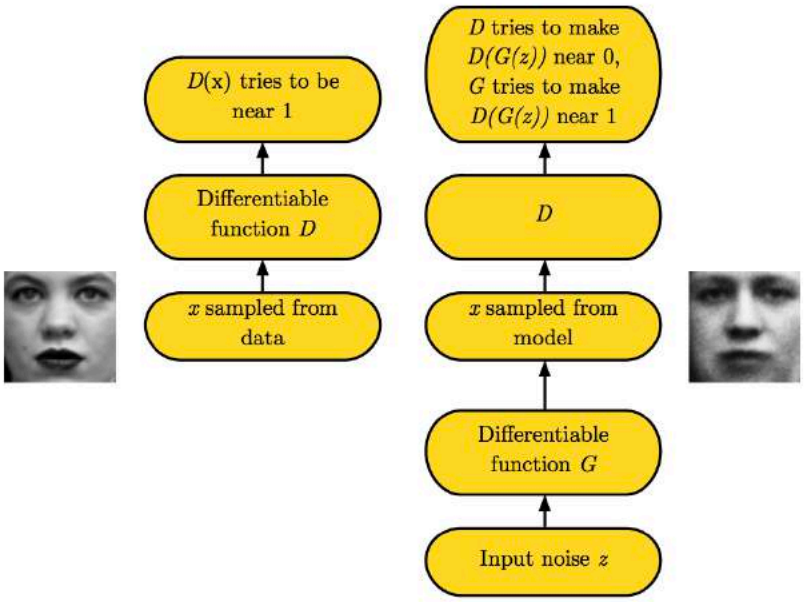

Fig 10. Adversarial training in Generative Adversarial Networks. [27]

In 2015 Alec Radford, et al. [**29**] introduced the combination of Convolutional Neural Networks and unsupervised learning in the so-called Deep Convolutional Generative Adversarial Networks (DCGAN). Trained in various image datasets, as LSUN bedroom dataset, model showed promising results in generating new bedroom images. Nowadays, most of GAN architecture are loosely based on DCGAN architecture.

The large variety of images Generative Adversarial can generate, makes this model specially useful in fields like art where there are more than one possible correct output, using an style transfer function to change the image from one domain to another [**30**]. Also generating new images from rough representation of an image made by users (iGAN) [**31**] and outperforming state-of-the-art super-resolution CNN architectures [**32**]. Moreover, Isola *et al.* [**33**] created a a new concept named image to image translation which encompasses applications as converting a satellite photo in a map or an sketch image into a realistic one.

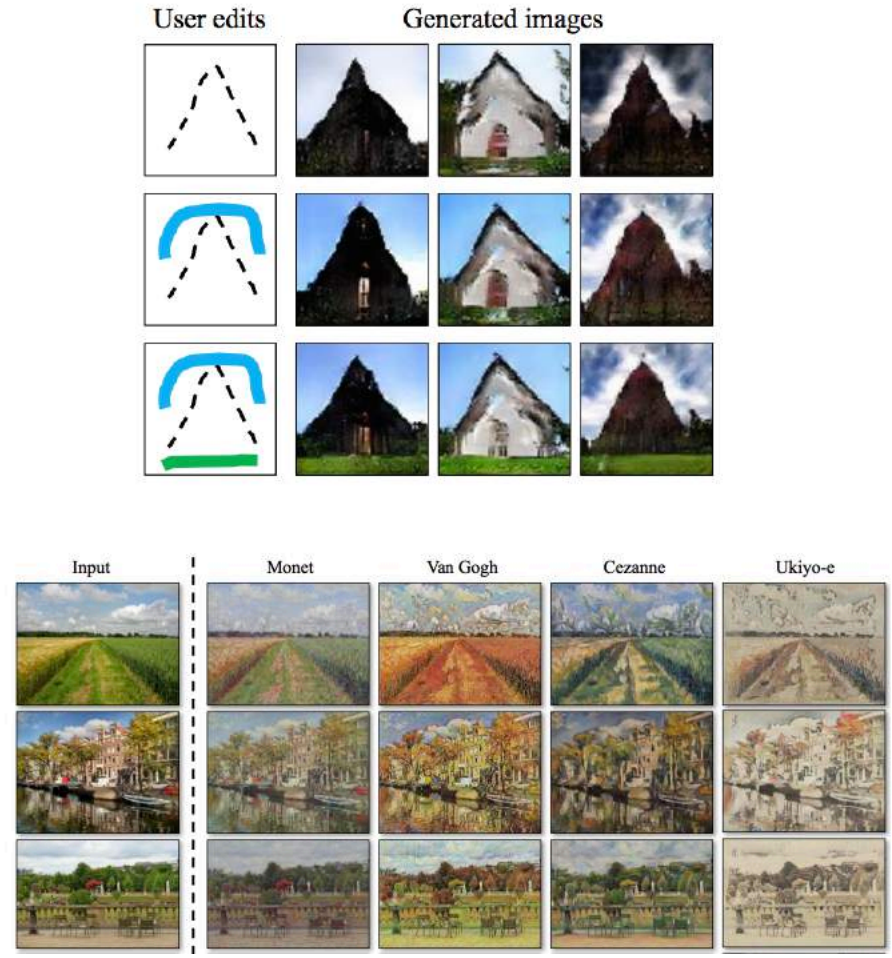

Fig 11. Interactive GAN [30] and CycleGAN [29] result examples. Top to bottom: iGAN / CycleGAN.

2.4 Image Quality Assesment

Image quality assesment is one of the most challenging issues in digital image processing systems. Image enhancement techniques need a metric to measure whether or not an enhnaced image is better than the original when evaluating a model. Human eye and subjective opinion are the best metric since we are the ultimate users of most media applications. However it is neither possible nor reasonable in automatic systems where hundreds images are processed. Also they are expensive and, since they are subjective, they do not opperate following an specific criteria, so results can be biased. Therefore, objective evaluation methods have been developed to model human vision system as closely as possible.

Mean Squarred Error (MSE) and Peak signal-to-noise ratio (PSNR) are the simplest and most widely used in this task [**34**] [**35**] [**36**]. MSE compares two image signals and provides a quantitative score of distortion between them (2) . MSE is also expressed as a PSNR of two images: reference *f* and enhanced *e* (3), where *L* is the

maximum value of a pixel corresponding to the image format. In RGB images, PSNR is a mean of each channel in gray scale, so L is 255. PSNR is measured in decibels (dB) and good quality values depends on the bit representation of the image. Though, photo enhancement has state of the art values, which analysed in further chapters, and are the ones used to compare the results.

$$MSE(f,e) = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}(f_{ij} - e_{ij})^2 \quad (2)$$

$$PSNR(f,e) = 10\ log_{10}\left(\frac{L^2}{MSE(f,e)}\right) \quad (3)$$

Nevertheless, PSNR does not take into account the structural information of the image. Human vision system extracts structural information of objects, scenes etc., when seeing an image. Thus, Structural Similarity Index (SSIM) was developed as "direct way to compare the structures of the reference and the distorted signals" [**36**].

SSIM is comprised of three independent components: luminance (l), contrast (c) and structure (s). SSIM values are measured between 0 and 1, being 1 the best quality result. Equation 4 shows SSIM composition when comparing a signal x and a distorted one y, and α, β, γ are parameters, greater than 0, which represents the importance of each component.

$$SSIM = [l(x,y)]^{\alpha} \cdot [c(x,y)]^{\beta} \cdot [s(x,y)]^{\gamma} \quad (4)$$

In image quality assesment, SSIM is called mean SSIM (MSSIM) to refer a to a single measure of the overall image. MSSIM of a reference image f and a enhanced one e is:

$$MSSIM(f,e) = \frac{1}{M}\sum_{i=1}^{M} SSIM(f_i\ ,\ e_i) \quad (5)$$

For our purposes, photo enhancement is measured using these metrics so models can be compared with each other. However, since photography, specially editing tasks, has a high level of subjectivity, subjective metrics such as single and double-stimulus [**37**] are also important measures when evaluating the results of a technique or a

learning-based model. In this study, non subjective methods such as surveys or user studies are not applied, but the obtained PSNR and SSIM values are contrasted with the enhanced visual results when selecting the model.

2.5 Photo Enhancement: Convolutional Neural Networks

Photo enhancement can be addressed in many different ways, but in general the main goal is, given an image *I* and an enhanced version *E*, generate a model capable to learn the mapping function between them.

Convolutional Neural Networks have been applied in photo adjustment. One of the first deep learning models to be applied in photo enhancement was developed in 2016 [**38**]. It aims to do automatic photo adjustment, which unlike traditional techniques, takes image content into consideration instead of perform globally. It was trained using 70 images from Flickr and its enhanced version retouched with Photoshop by a professional photographer. Superpixel method was applyed for training, so they have more than a million samples taken from the 70 pairs of images. They introduced semantic analysis of image content using object recognition algorithms. Three models with exciting results were obtained performing 3 different enhancement styles.

Gharbi *et al* [**39**] proposed a full convolutional model for real time image enhancement. They provide a faster enhancement in high resolution images by extracting features in a low level resolution, then changing parameters to enhance the image and finally upsampling the result (bilateral learning). There are two branches in this convolutional architecture, so local and global features can be extracted separately. Various models were trained using paired data (original - enhanced) of different filters as Local Laplacian, Face brightening …, and also in benchmark MIT-Adobe 5K dataset [**40**] composed by 5000 images each of them retouched by 5 different photographers. Bilateral learning performs good in terms of time as they obtained a faster enhancement compared with other state of the art models at the time, but enhancement results were just another approximation of existing filters.

Approaches in image colorization using CNN as [**41**] also rised. This issue tackles color restoration in gray scale images. It showed good results using a 16 layer convolutional network outperforming other methods. Also low-light image enhancement is a very common issue which have been addressed by authors in [**42**]. The technique is focused in enhancing low contrast images, and they worked using a convolutional network architecture which learned an end-to-end mapping between low light and bright images. However, these two approaches and others like, denoising [**43**] and deblurring [**44**] tasks which aims to remove artificial noise and blur added to images, respectively, can be considered as sub-problems inside photo enhancement issue. Therefore, a combination of image adjustment with these other techniques yield better global image enhancement.

Recent Generative Adversarial architectures outperform some of this CNN models, presenting more powerful models but they also showed instability problems during the training process. Nevertheless, results are encouraging and good enough to be examined.

2.6 Photo Enhancement: Generative Adversarial Networks

The emergence of Generative Adversarial Networks make researchers takcle image/photo enhancement as an image to image translation problem, where a network architecture can learn function to map the translation from one image domain into other image domain. In his work, Isola *et al* [**33**] generalise the concept of image-to-image translation and define automatic image-to-image translation "*as the task of translating one possible representation of a scene into another, given sufficient training data.* ". This work encompass previous investigation on image enhancement made with CNN and also gives a new line of investigation which may yield numerous new applications.

The proposed architecture is a variant of baseline GAN architectures called Conditional GANs. The main difference with normal GANs is that instead of just giving a random noise vector to the Generator and a generated sample to the Discriminator as explained before, conditional GANs learn a mapping given an input image (x) and a noise input vector (z) for the Generator, and the same input image (x) and the generated sample ($G(x,z)$) for the Discriminator. In this way, both Generator and Discriminator are conditioned by x, so Generator is not only focused to fool Discriminator, but to generate a sample near the ground truth x.

The model was trained using image patches instead whole images, which showed effective results. Performance of this architecture was tested training models on different datasets, so each of them is conditioned by a certain task. For example translation from segmentated images, taken from Cityscapes dataset, to a photogrpah, a map, taken from Google maps, to an aerial photograph or black and white images, taken from ImageNet dataset, to colored ones, among others. The results were evaluated using FC-Score which aims to classify the generated images using a classifier trained in ImageNet dataset, having a better FC-Score as better is the classification accuracy.

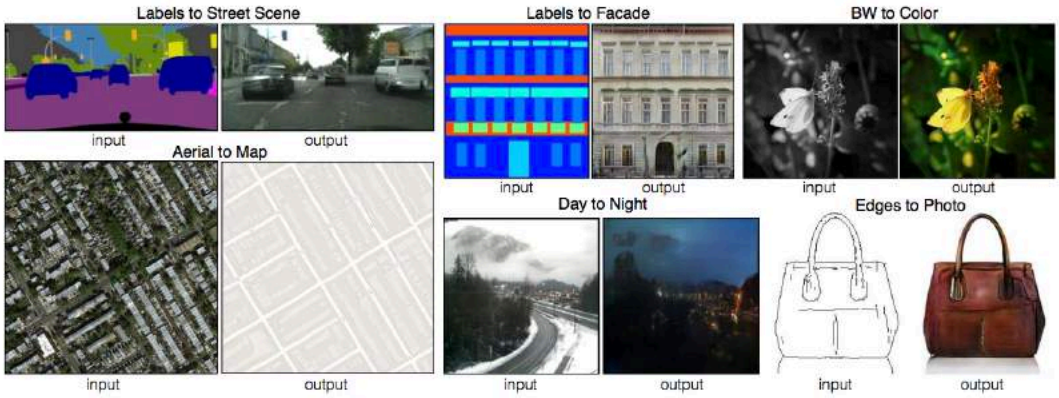

Fig 12. Conditional Adversarial GAN results. [33]

After, Ignatov *et al* [**45**] took another approach for image-to-image translation by learning a mapping between photographs taken with a mobile phone and photographs taken with a DSLR camera. The model is a combination of CNN and GAN architectures. Figure 12 shows the overall architecture. The image enhancement network is fully convolutional with four residual blocks which mix convolutional and batch normalization layers. Below image enhancement network, there is a Discriminator network which is trained to distinguish between phone and DSLR camera images, and it outputs the probability that an image is taken by a DSLR camera. The peculiarity of this model remains on its particular loss function. It is defined as a weighted sum of 3 functions which takes into account three important elements on image quality: color, content and texture. Authors pointed that choosing a losss function based on differences pixel-to-pixel it's not a accurate approach due to distornsions produced by different cameras or devices which yield a non-constant shift of pixels. The dataset (DPED dataset) was originally made by the authors and is composed by set of images taken simuntaneously with 4 different devices: a *Blackberry Passport,* an *iPhone 3GS*, a *Sonny Xperia Z*, and a *Canon 70D* DSLR camer. 22K images were taken in total. As Isola *et* al, they trained their models using patches. Three models were trained in total, each of them using the phone-DSLR camera image pairs as the training set. Quantitative evaluation of the models was made using PSNR and SSIM, but also a subjective method was proposed to check whether the model is useful for users. The model got very good feedback from subjective evaluation due to users choose the enhanced image over the original one, and DSLR or enhanced indistinctly.

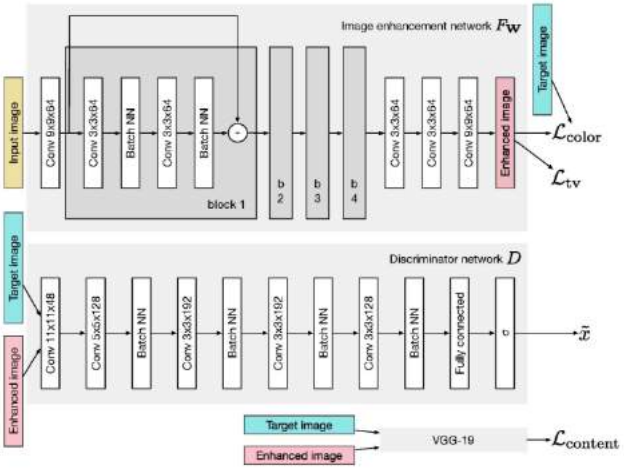

Fig 13. DPED archiecture overview. [45]

However, there are some flaws in the model as too much contrast level or noise amplification due to GAN architecture. Also, the need of strong supervision for training the model (need of original phone image and target DSLR camera image pairs) bring them to the tedious process of generating their own dataset, which is a widespread problem in image tasks. Therefore, thanks to how Isola *et al* approached the problem and the results of Ignatov *et al*, a "weakly-supervised approach" was later proposed.

Weakly Supervised Photo Enhancer (WESPE) [**46**] is an improved version of DPED proposed by the same authors. The main goal was to obtain same visual results as in previous work but in a more unsupervised manner. Instead of using before-after image pairs, this model uses two independent datasets: one containing photos from a mobile, and another one containing arbitrary high quality images. In this way, they can generalise the model for any kind of camera taking advantage of recent advancements by Isola *et al.* in combination with their previous loss function which yielded photorealistic quality results. The proposed GAN architecture has the peculiarity of having two discriminators and, of checking consistency in generator's network. Both discriminators aims to differentiate between high-quality image and the enhanced one, but one of them is based on the color and the other on the content, which makes Generator have more difficulties to fool discriminator. Generator consistency is checked by mapping input image *x* in the target domain *Y*, and the other way around, so they ensure generator preserves *x's* content, and avoid image pairs during training. To achive that, two networks were used: *G* for generative process and *F* for inverse generative process.

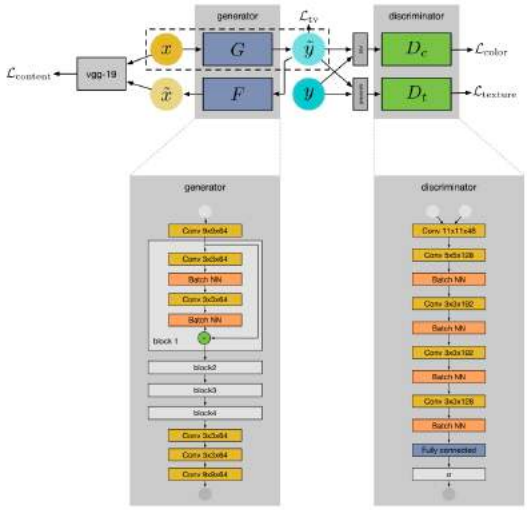

Fig 14. WESPE architecture overview. **[**46**]**

The model was trained on the DIV2K dataset [**47**] used to tackle super-resolution task and contains 1000 high resolution images. Results in other datasets as Cityscapes or KITTI showed results that compete with the supervised version of the model.

Along the same line of unpaired learning, a team from National Taiwan University has recently developed an unpaired photo enhancer model [**48**]. The proposed model is a mixture of two GAN architectures: Wasserstein-GAN (WGAN) and CycleGAN. Wasserstein-GAN [**49**] provides a solution for training instability problems in generative adversiarial architectures showing efficient training results when using Wasserstein distance as the loss function, which in general terms, calculates the minimum cost of transforming a probabilistic distribution *P* into other probabilistic distribution *Q*. CycleGAN [**30**] is used to check cycle consistency between original and target domains (original photo domain and enhanced photo domain), which, as in Weakly Supervised Photo Enahncer, is the solution to avoid paired training.

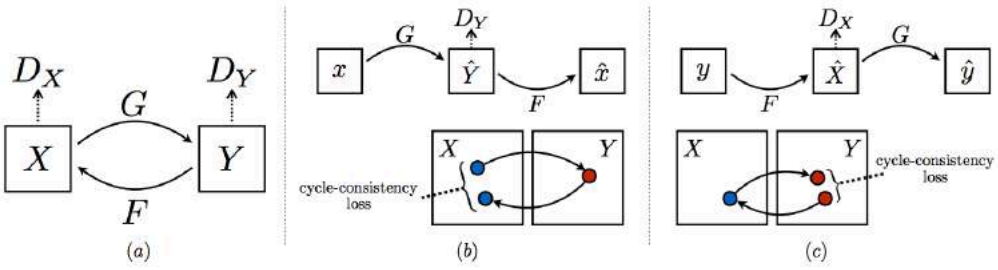

Fig 15. Mapping functions and consistency in CycleGAN. **[**30**]**

Figure 16 shows the overall architecture. On the right hand side, Generator network is presented. It is based on U-Net architecture [**50**] which originally was created for biomedical image segmentation, and residual learning [**51**]. U-Net follows same contraction process to extract image features as in CNN, but it is supplemented with an expansive process which upsamples the feature maps to reconstruct the image again. The contraction and expansive processes are concatenated with each other in order to preserve image content while reconstructing the image. However, the original architecture does not perform well in photo enhancement due to its lack of ability to extract global features. In the proposed architecture, authors modified the original from the fifth contraction layer, where the gobal features are extracted. Residual learning helped in better learning process as it showed in other image processing applications, and also in faster convergence during training process.

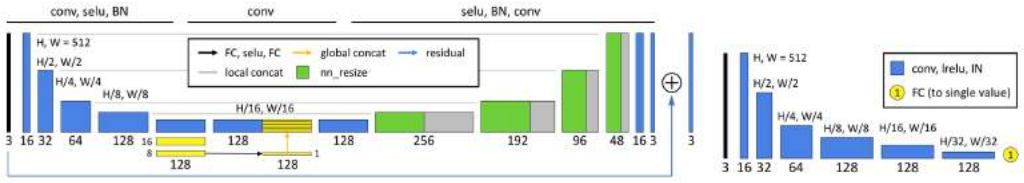

Fig 16. Unpaired photo enhancement architecture overview. **[**48**]**

The proposed architecture only needed one training set containing the images with the desired characteristics, so the model learns to generate images with the same distribution of enhanced images directly.Three models were trained using two different datasets (Table II): MIT-Adobe 5K dataset, and another one extracted from Flickr composed by 627 well ranked HDR photographs.

TABLE II

SUMMARY OF LPGAN MODELS

MODEL	**TRAINING DATASET**	**PAIRED LEARNING**
LPGAN_736	Flickr HDR	NO
LPGAN_577	MIT-Adobe 5K	NO
LPGAN_604	MIT-Adobe 5K	YES

Results are compared with 5 other works as DPED described before, and also with a supervised version of the developed model, using a user study. Visual results show that there are less difference between unsupervised and supervised learning results, which places unpaired learning in a better position, and PSNR and SSIM values outperform the rest of the models. Moreover, user study showed that the model trained using HDR photographs has the best results among the rest.

3. MODEL SELECTION

Once the review of different photo enhnancement approaches is done, now it's time to evaluate their experimental results in order to choose the best architecture. Table III shows the chosen models, their architecture and whether or not it's been possible to find an implementation.

As it can be seen, full convolutional based architectures have been discarded because, in general, the models tackle photo enhancement as a specific problem of image colorization, image denoising, image deblurring… Automatic Photo adjustment and bilateral learning models are the only ones which have comparable results, however it has not been possible to find their implementation.

Moreover, as it was discussed before, GAN architectures are a more innovative approach which has shown great results in image processing tasks, and the developed photo enhancement models address the problem in a more global manner.

TABLE III

SUMMARY OF BEST CNN AND GAN MODELS

MODEL	ARCHITECTURE	PRE-TRAINED MODEL
DPED	CNN-GAN	YES
WESPE	GAN	NO
LPGAN_736	GAN	YES
LPGAN_577	GAN	YES
LPGAN_604	GAN	YES

Weakly Enhancer model, WESPE, as it is discussed above, has interesting results using unsupervised learning, though, project's website do not share any code or pre-trained model to evaluate its results. Despite this, the implementation for DPED and the four LPGAN's models is shared by the original authors, so in next section experimental results will be discussed.

3.1 Pre-trained models and experimental environment

3.1.1 What is a pre-trained model?

In order to evaluate a model, first it's necessary to find an implementation of the model or build one from scratch. In the field of neural networks, there are plenty of programming languages and libraries to help users develope learning models (R, Matlab, Python, Torch, Pytorch, Tensorflow, Keras…). Python, programming language, and Tensorflow, open source machine learning library developed by Google, are two of the most widely used softwares in the developement of deep learning models.

Tensorflow [**52**] allows training deep neural networks in combination with Python. As it is explained in section 2, learning process it is an iterative method where at each iteration, or epoch, the model is fed with a set of data samples, extracted from training set, in order to adjust its parameters. Moreover, if training data are images the number of parameters grows large and fine adjustment of weights may take hours. Tensorflow provides functions to take a snapshot of a model at each training epoch so one can save and evaluate the model keeping the weights values in a specific epoch. In Tensorflow, this is called **checkpoint**, and is very helpful because sometimes training process is not stable and training accuracy may vary, so one may want to go some steps back and recover previous weight values. It is also useful in case one wants to share the best performance of the model with other people, so it can be used without having to retrain the network again, and that is what is called, a **pre-trained model**.

3.1.2 Pre-trained models

As it is mentioned before, DPED and LPGAN models are the ones which can be experimentally evaluated. The models are implemented using Python and Tensorflow, and for each model in Table IV authors have made available a tensorflow checkpoint i.e. pre-trained model: DPED checkpoint extracted from [**53**] and the four LPGAN checkpoints extracted from [**54**].

TABLE IV

SUMMARY OF EVALUATED PRE-TRAINED MODELS

MODEL	TRAINING DATASET	PAIRED LEARNING
DPED	DPED dataset	YES
LPGAN_736	Flickr HDR	NO
LPGAN_577	MIT-Adobe 5K	NO
LPGAN_604	MIT-Adobe 5K	YES

The implementation of each model has been adapted so it can be executed in the specific environment presented below. After adapting the models, they run in Tensorflow 0.12 and 1.12, and Python 2.7 or 3.6. The code is available at the following repository:

https://github.com/fromage2/Mix-of-photo-enhancement-models-s-evaluation .

3.1.3 Experimental environment

The evaluation of the pre-trained models mentioned above are subject to computational limitations. Table V shows the technical specifications of the laptop used to run the evaluation.

TABLE V

EXPERIMENTAL ENVIRONMENT HARDWARE SPECIFICATIONS [55]

Laptop model	Lenovo ThinkPad X270
CPU	Intel Core i5-7200U 2.5 GHz
RAM	8GB
GPU	Intel HD 620

Models are **evaluated within two different test sets**: DPED test set and a random mobile photographs test set (RAMP). On one hand, DPED test set is selected because experimental results of both DPED and LPGAN models are evaluated within it, so it is a good way to check whether the results are feasible. On the other hand, a random mobile photographs test set is used to ensure an evaluation with unseen data and also, since the intention is to deploy the model in a mobile device, the photographs have been taken with different mobile phones to simulate the most realistic possible scenario.

There is also a scaled down version of both data sets due to better comparison between models. LPGAN models are trained using 512x512 images and it only works with images of that fixed resolution. Since DPED allows images of lower dimensions, it can be compared with LPGAN by just downsampling test images. In the down scaled test sets, original images have been resized so the longer side (height or width) is 512, and the other is rescaled using the scaled factor associate so image does not lose its appearance.

$$Scale\ factor\ = \frac{512}{longerSide} \tag{6}$$

Both test sets with their corresponding scaled down version, are available online at the following: https://github.com/fromage2/Test-sets-used-for-evaluating-photo-enhancement-models .

1. *DPED test set*: extracted from DPED project, this test set is divided in 3. Each part contains 29 photographs taken with one of the three mobile phones (iPhone, Blackberry, Sony) which are a subset of the ones used to evaluate the model in the original paper [**45**]. Since there are 3 different models the whole test set is composed by 87 images. The three sets of 29 images contains the same photographs, but, since they were taken with different mobile phones the difference resides in their resolution.

2. *RAMP test set*: this is a homemade test set composed by 13 photgraphs taken wih different mobile devices as iPhone 6, iPhone 7 and BQ Aquaris V.

As it was discussed in section 2, the **metrics to evaluate the results** are SSIM and PSNR. PSNR implementation is very simple and there are plenty of functions online which calculate it, so two implementations were taken to ensure a good evaluation [**56**]. SSIM is a bit more complex function, but the open source package *scikit-image* [**57**] provides a collection of image processing algorithms including SSIM implementation [**58**]. Apart from the obtained PSNR and SSIM values, visual results are also taken into account becuase not always numerical values line with image appearance.

Moreover, for finally selecting the model, CPU time is also measured since the model has to be as efficent as possible so it might fit better in mobile devices.

3.2 Pre trained models evaluation

3.2.1 DPED models

As it was mentioned, DPED has three different models, one for each mobile phone iPhone, Blackberry and Sony. Table VI shows results on DPED test set with original image dimensions. Image size of each model is in line with the resolution of the photographs taken with each mobile. PSNR/SSIM paper and PSNR/SSIM experiment are the values which appears in the original paper and the mean from our test, respectively. Experiment values deviate from the ones presented in the original paper, but at least Sony model shows to has the highest scores in both paper and experimental results. Visual results shows good performance of the models.

TABLE VI

DPED MODELS MEAN PSNR AND SSIM RESULTS EVALUATED IN DPED TEST SET ORIGINAL RESOLUTION VERSION.

MODEL	IMAGE SIZE	PSNR (paper)	PSNR (experiment)	SSIM (paper)	SSIM (experiment)
DPED_iphone	2048 x 1536	20.08	15.15	0.9201	0.7682
DPED_blackberry	2080 x 1560	20.07	20.48	0.9328	0.8593
DPED_sony	1680 x 1260	**21.81**	**26.08**	**0.9437**	**0.9226**

In Table VII results on scaled down DPED test set are presented. Here the values of the mean PSNR and SSIM are lower than in original original dimensions data set except for iPhone model, but Sony model is still the best. However, the small variation of PSNR and SSIM does not match the huge variation in the visual results. This might happened because each model were trained using an specific resolution and lower dimensions may produce distorsion and blurry images.

TABLE VII

DPED MODELS MEAN PSNR AND SSIM RESULTS EVALUATED IN DPED TEST SCALED RESOLUTION VERSION.

MODEL	IMAGE SIZE	PSNR (experiment)	SSIM (experiment)
DPED_iphone	512 x 384	16,42	0,7959
DPED_blackberry	512 x 384	22,27	0,8681
DPED_sony	512 x 384	**24,66**	**0,8775**

Fig 17. Results DPED models evaluated in DPED test set original resolution version. Left to right, top to bottom: Original Sony image / DPED_Sony / Original Blackberry image / DPED_Blackberry / Original iPhone image / DPED_iPhone

Fig 18. Results DPED models evaluated in DPED test set scaled resolution version. Left to right, top to bottom: Original Sony image / DPED_Sony / Original Blackberry image / DPED_Blackberry / Original iPhone image / DPED_iPhone.

Finally, Table VIII shows results on RAMP test set. As it was expected, the PSNR and SSIM mean values are similar to the ones obtained in scaled DPED test set, and visual results are also blurry.

TABLE VIII

DPED MODELS MEAN PSNR AND SSIM RESULTS EVALUATED IN RAMP TEST SET SCALED RESOLUTION VERSION

MODEL	IMAGE SIZE	PSNR (experiment)	SSIM (experiment)
DPED_iphone	512 x 384	16,33	0,8460
DPED_blackberry	512 x 384	22,64	0,9000
DPED_sony	512 x 384	**23,79**	**0,9187**

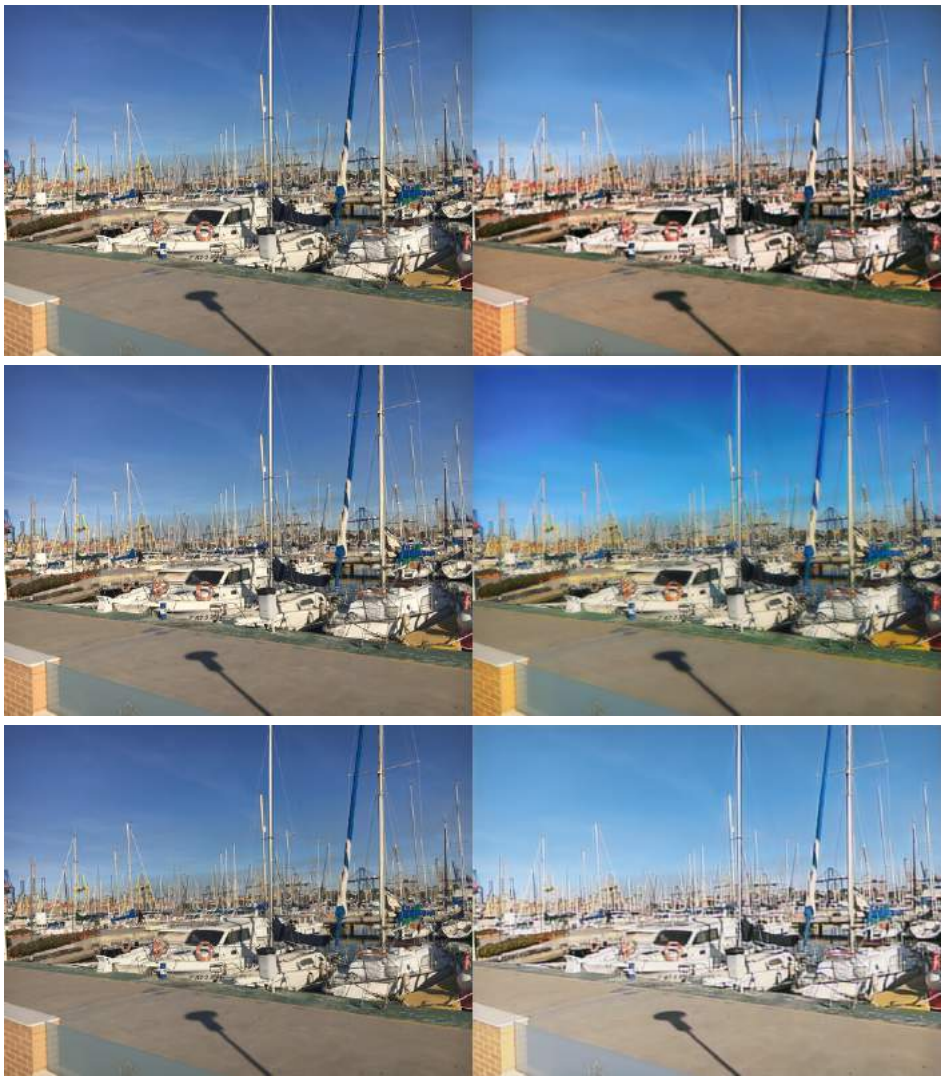

Fig 19. Results DPED models evaluated in RAMP test set scaled resolution version. Left to right, top to bottom: Original RAMP image / DPED_Sony / Original RAMP image / DPED_Blackberry / Original RAMP image / DPED_iPhone.

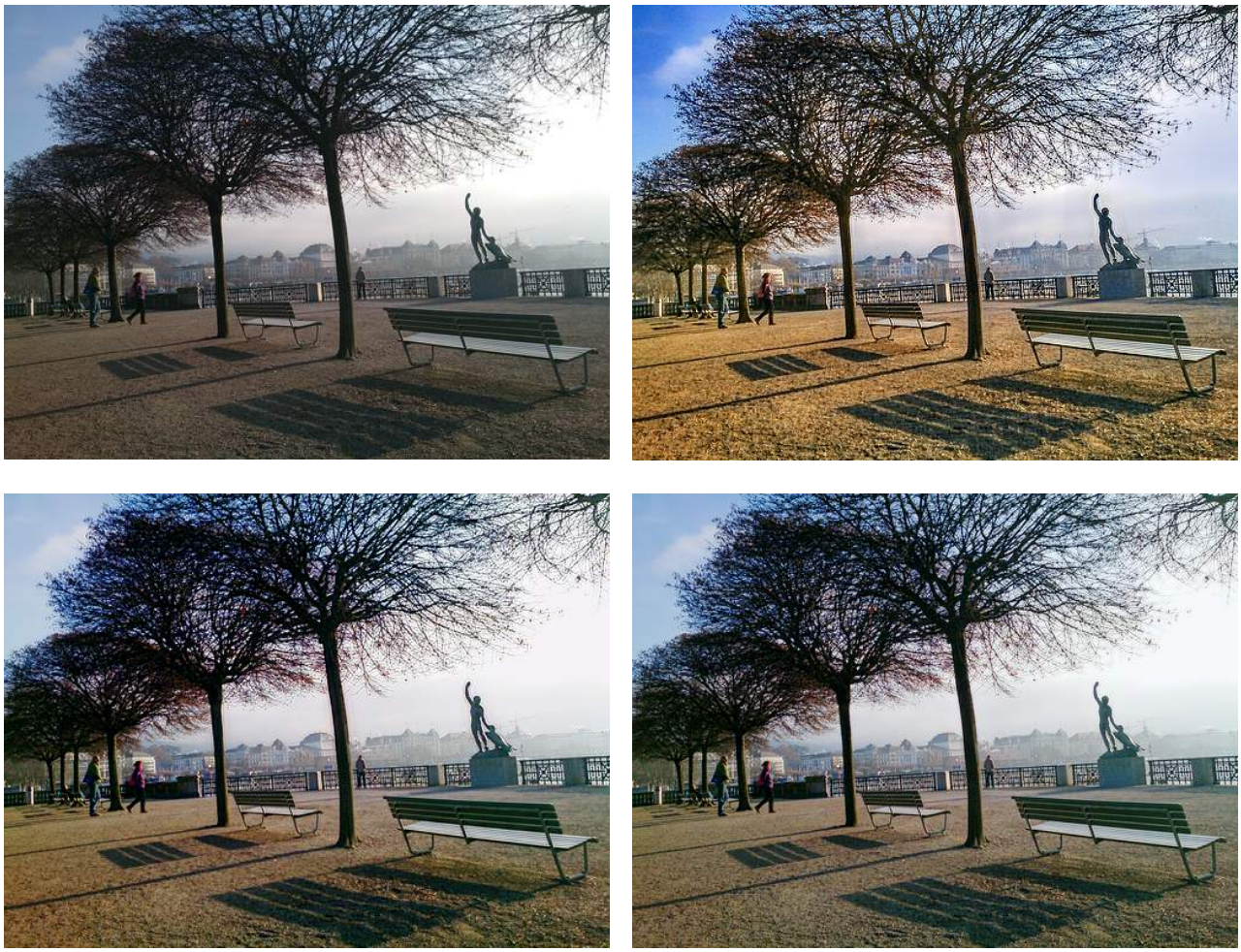

Fig 20. Results LPGAN models evaluated in DPED test set scaled resolution version. Left to right, top to bottom: Original DPED image / LPGAN_736 / LPGAN_577/ LPGAN_604.

In this case, the evaluation in RAMP test set (Table X) yield similar results as in previous test set. LPGAN_604 is again the most valued model in terms of PSNR and SSIM, and visually images keep the enhancement style of each model.

After comparing the results, it can be seen that even if supervised model outperform the other two in terms of numeric values, unsupervised models have pretty good visual results which compete with the supervised one.

TABLE X

LPGAN MODELS MEAN PSNR AND SSIM RESULTS EVALUATED IN RAMP TEST SET SCALED RESOLUTION VERSION

MODEL	IMAGE SIZE	PSNR	SSIM
LPGAN_736	512 x 384	20,43	0,7915
LPGAN_577	512 x 384	20,94	0,8813
LPGAN_604	512 x 384	**22,09**	**0,9071**

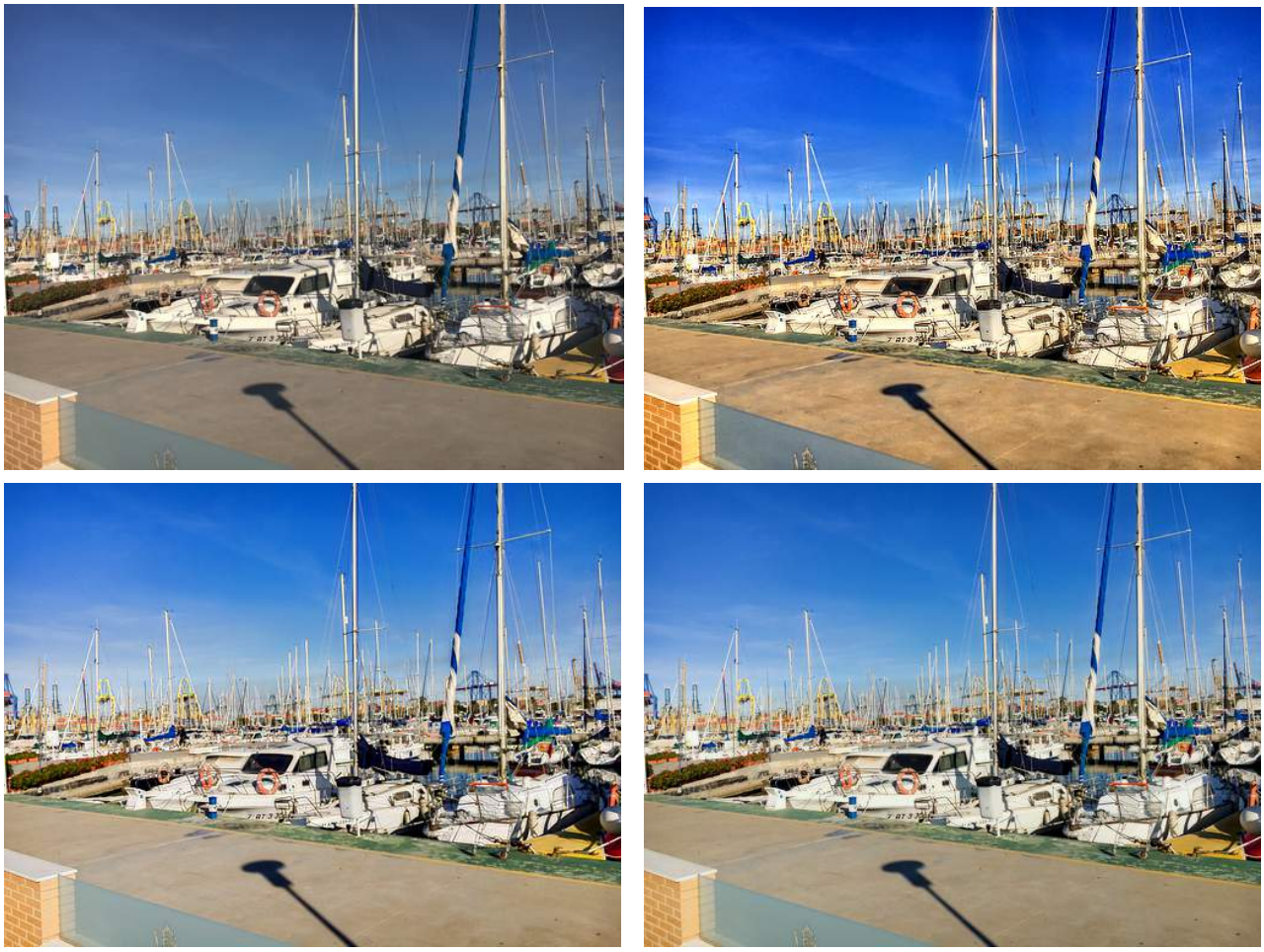

Fig 21. Results LPGAN models evaluated in RAMP test set scaled resolution version. Left to right, top to bottom: Original RAMP image / LPGAN_736 / LPGAN_577/ LPGAN_604.

Therefore, results showed that DPED_sony model have the best values in the three test sets. In general, the three models obtained PSNR and SSIM values according to the original peper when enhancing images with same resolution as the training set and visual results support those values. However, when image resolution change, PSNR and SSIM decrease, and probably, the strong supervision of the training process leads to a low image quality results.

3.2.2 LPGAN models

As it was mentioned before, LPGAN models are only evaluated in the scaled down versions of the test sets. Table IX shows results in DPED test set, which are similar to DPED models. In the original paper, LPGAN_736 is the one compared with DPED model because in their study this model obtained the highest score. However, our experimental values variate from the original ones, and supervised model LPGAN_604 has the best PSNR and SSIM values, which also differs from original paper results.

TABLE IX

LPGAN MODELS MEAN PSNR AND SSIM RESULTS EVALUATED IN DPED TEST SET SCALED RESOLUTION VERSION.

MODEL	**IMAGE SIZE**	**PSNR (experiment)**	**SSIM (paper)**	**PSNR (experiment)**	**SSIM (experiment)**
LPGAN_736	512 x 384	23,8	0,900	20,53	0,7544
LPGAN_577	512 x 384	-	-	20,70	0,8872
LPGAN_604	512 x 384	-	-	**22,73**	**0,9051**

As it can be seen in Figure 20, there are differences between LPGAN_736 model enhancement style and the other two models. This happened because LPGAN_736 is trained using HDR images from Flickr, which are usually high contrasted and more colorful images. LPGAN_577 and its supervised version, LPGAN_604, are trained using images retouched by a professional photographer, which uses a different enhancement style.

3.3 Model selection

After evaluating the image quality of the models, numerical results show that Sony has the best performance but LPGAN_604 and LPGAN_577 are the only one which exceeds SSIM values. However, visual results demonstrate that numerical metrics not always reflect the real model's performance.

TABLE XI

SUMMARY MEAN PSNR AND SSIM RESULTS OF DPED AND LPGAN MODELS

	Scaled DPED test set		*Scaled RAMP test set*	
MODEL	**PSNR**	**SSIM**	**PSNR**	**SSIM**
LPGAN_736	20,53	0,7544	20,43	0,7915
LPGAN_577	20,70	0,8872	20,94	0,8813
LPGAN_604	22,73	**0,9051**	22,09	0,9071
DPED_iphone	16,42	0,7959	16,33	0,8460
DPED_blackberry	22,27	0,8681	22,64	0,9000
DPED_sony	**24,66**	0,8775	**23,79**	**0,9187**

In order to select the model, CPU time is measured. Table XII shows the results, and as it can be seen, LPGAN models are more than two times faster than DPED ones. Measurements are done using scaled data sets so images have same resolution, and in the same laptop, so time efficiency resides in model architecture.

In general terms, DPED_Sony has the best image quality performance in high resolution images though the highest execution time, even in scaled down images. LPGAN_604 have good performance in scaled images and the execution time is one of the best. However, as it was mentioned before, unsupervised models (LPGAN_736 and LPGAN_577) yield encouraging results which compete with supervised models. Evaluating CPU time, the two unsupervised models produced the quickest results. Therefore, as the objective of this thesis is also to choose an efficient model suitable for mobile devices, LPGAN_736 and LPGAN_577 are the selected models to develop the photo enhancer app.

TABLE XII

MEAN CPU TIME RESULTS OF DPED AND LPGAN MODELS EVALUATED IN DPED AND RAMP SCALED TEST SETS

	CPU time (ms)	
MODEL	*Scaled DPED test set*	*Scaled RAMP test set*
LPGAN_736	**606,62**	**348,42**
LPGAN_577	764,20	706,22
LPGAN_604	791,37	783,95
DPED_iphone	2.451,90	1.097,52
DPED_blackberry	2.397,59	2.306,06
DPED_sony	2.787,69	2.248,40

5. MOBILE APPLICATION

Once models are selected now it's time to evaluate whether or not they are suitable to be executed in a mobile device. In order to do that, models have to be optimized and converted to an specific format so they can be launched in a mobile application.

5.1 Model conversion

First option is Qualcomm *dlc* format, which uses Snapdragon Neural Processing Engine SDK [**59**]. It is suitable with Snapdragon processors which are the ones used in Android devices nowadays. However, Qualcomm is a relatively new format and it is not yet developed to support any kind of layer when using neural networks [**60**].

Since the attempt to convert the two selected models into Qualcomm format failed, other approach is applied using Tensorflow Mobile library [**61**]. Tensorflow's network achitectures are composed by nodes, which are units where the opperations of the network take place. Some of these operations are only used during the training process and afterwards never used anymore. Tensorflow Mobile provides functions to optimize model architectures and prune the nodes which are not necessary in the enhancement process. The resulting model is called *frozen model* and it is the one which is executed in the mobile device when enhancing an image. Unlike what happened with Qualcomm, selected models (LPGAN_736 and LPGAN_577) could be succesfully converted to their frozen counterpart.

5.2 Photo enhancer app: environment and structure

Photo enhancer app is the mobile application to enhance images using the frozen models previously mentioned. The app has been developed using Android Studio, and it is structured in two simple java classes.

The first class (MainActivity) deploys the visual content and controls the flow of the app. As it can be seen in Figure 22 there are two buttons: one to choose a photo from the gallery, and the other to enhance the chosen picture. This class also preprocess the input image to scale it down, so image's larger side is maximum 512, as it is explained in section 3. After that, as the two selected models can only process 512x512 images, padding is added to the shortest side of the image to finally get the required resolution. When the image is enhanced the padding is removed and the enhanced image finally remains in its scaled down version.

The other class (ActivityInference) perform the enhancement process itself. It loads the frozen model, and receives the preprocessed image from MainActivity to enhance it.

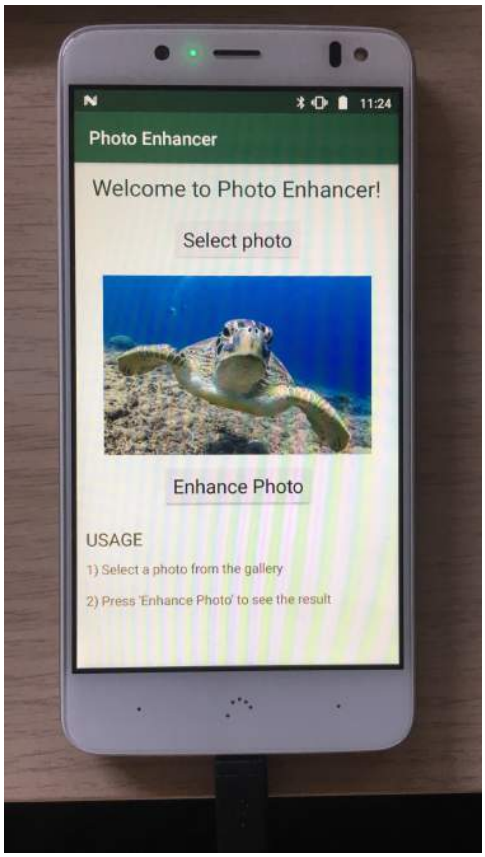

Fig 22. Photo enhancer app overview

5.2 Frozen model evaluation and selection

In order to select the best frozen model, an evaluation of LPGAN_736 and LPGAN_577 frozen models is done. The evaluation consists in enhancing 10 images taken from both DPED and RAMP test set, using the *photo enhancer* app mentioned before. PSNR, SSIM and CPU time values are again used as objective metrics to compare the models, but also visual results to contrast them. All values are extracted from the execution of the app in a BQ Aquaris V device. Table XIII and XIV shows the values of the metrics for each image used for testing in LPGAN_736 and LPGAN_577 frozen models, respectively.

TABLE XIII

PSRN, SSIM AND CPU TIME LPGAN_736 FROZEN MODEL OF 10 IMAGES EXTRACTED FROM SCALED DPED AND RAMPS TEST SETS. (Visual results see appendix)

Scaled DPED test set				*Scaled RAMP test set*			
Test image	**PSNR**	**SSIM**	**CPU Time (ms)**	**Test image**	**PSNR**	**SSIM**	**CPU Time (ms)**
1_DPED	21,80	0,8405	6510	*1_RAMP*	19,54	0,7668	6311
2_DPED	19,11	0,6900	6353	*2_RAMP*	19,19	0,7184	6213
3_DPED	21,06	0,6721	6173	*4_RAMP*	19,55	0,7586	6356
4_DPED	20,29	0,6903	6298	*7_RAMP*	19,99	0,7499	6323
5_DPED	20,98	0,7072	6306	*8_RAMP*	20,93	0,7682	6349
6_DPED	20,03	0,7138	6499	*9_RAMP*	19,08	0,6566	6368
7_DPED	19,43	0,7064	6328	*10_RAMP*	20,01	0,6830	6325
8_DPED	20,37	0,6650	6331	*11_RAMP*	21,47	0,7892	6274
9_DPED	20,01	0,6704	6362	*12_RAMP*	20,47	0,7098	6367
10_DPED	19,90	0,7078	6110	*13_RAMP*	19,89	0,7360	6473

TABLE XIV

PSRN, SSIM AND CPUE TIME LPGAN_577 FROZEN MODEL OF 10 IMAGES EXTRACTED FROM SCALED DPED AND RAMPS TEST SETS. (Visual results see appendix)

Scaled DPED test set				*Scaled RAMP test set*			
Test image	**PSNR**	**SSIM**	**CPU Time (ms)**	**Test image**	**PSNR**	**SSIM**	**CPU Time (ms)**
1_DPED	18,34	0,8483	7510	*1_RAMP*	20,04	0,8983	6891
2_DPED	20,88	0,7904	7353	*2_RAMP*	19,69	0,8404	6793
3_DPED	20,04	0,7844	7173	*4_RAMP*	20,05	0,8344	6936
4_DPED	21,60	0,8484	7298	*7_RAMP*	20,49	0,8984	6903
5_DPED	18,69	0,8544	7306	*8_RAMP*	21,43	0,9044	6929
6_DPED	19,64	0,8560	7499	*9_RAMP*	19,58	0,9060	6948
7_DPED	22,12	0,8013	7328	*10_RAMP*	20,51	0,8513	6905
8_DPED	19,80	0,8254	7331	*11_RAMP*	21,97	0,8754	6854
9_DPED	23,16	0,8262	7362	*12_RAMP*	20,97	0,8762	6947
10_DPED	20,36	0,8477	7110	*13_RAMP*	20,39	0,8977	7053

Table XV shows the mean of the results presented before. The values of PSNR and SSIM are almost the same. Visual results in Figure 24 confirm similarities in PSNR and SSIM values between models, though the enhancement style is not the same. Moreover, LPGAN_736 performs better in terms of time, so it is the chosen one to be used in *photo enhancer* app. Code implementation of the app is available at teh following repository, which also includes LPGAN_577 frozen model in case anyone wants to try it: https://github.com/fromage2/Photo-enhancer-app-

Figure 23 shows photo enhancer app results before and after the enhancement process. The image presented might not be perceived as in real life, but the enhanced image is better in a mobile screen than in the computer.

TABLE XV

MEAN PSNR, SSIM AND CPU TIME FROZEN LPGAN_736 AND LPGAN_577.

	Scaled DPED test set			*Scaled RAMP test set*		
MODEL	**PSNR**	**SSIM**	**CPU Time (ms)**	**PSNR**	**SSIM**	**CPU Time (ms)**
Frozen LPGAN_736	20,30	0,7064	**6.327**	20,02	0,7337	**6.321**
Frozen LPGAN_577	20,44	0,8264	7.127	20,53	0,8872	6.983

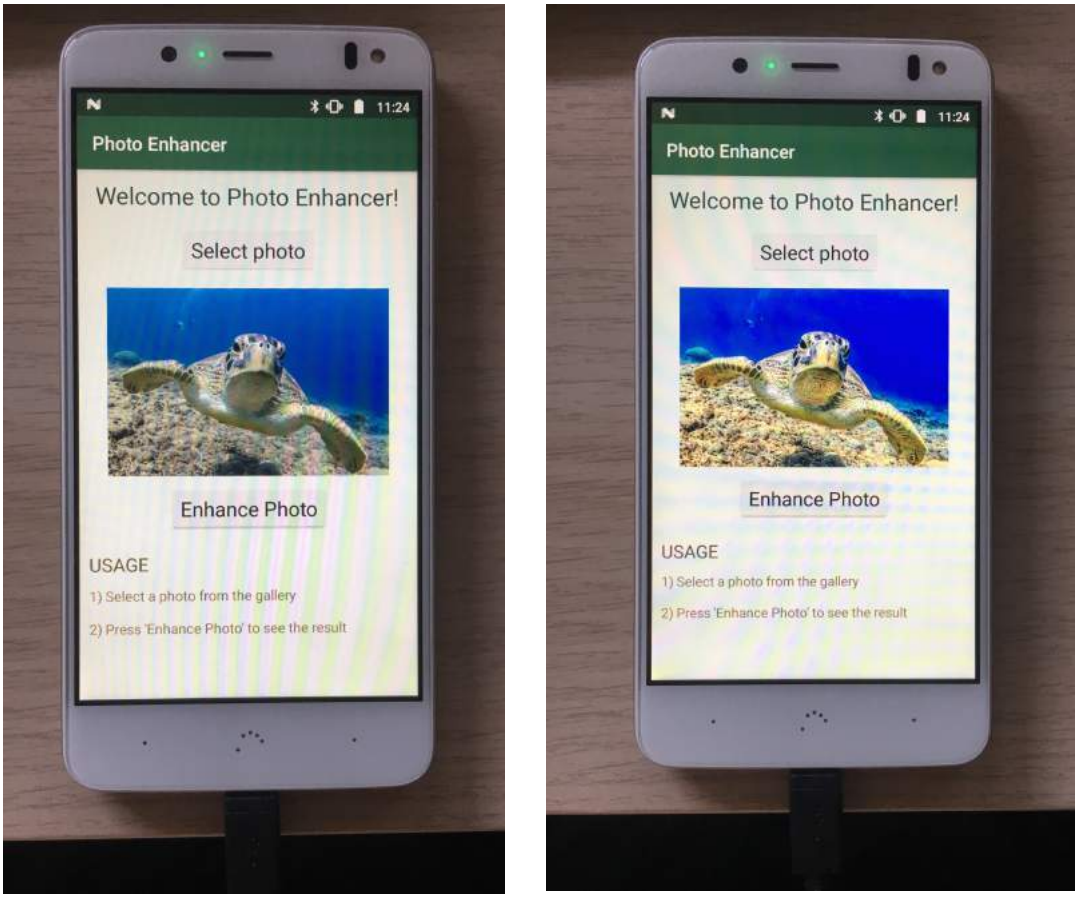

Fig 23. Before and after enhancement in Photo enhancer app, using LPGAN_736 model

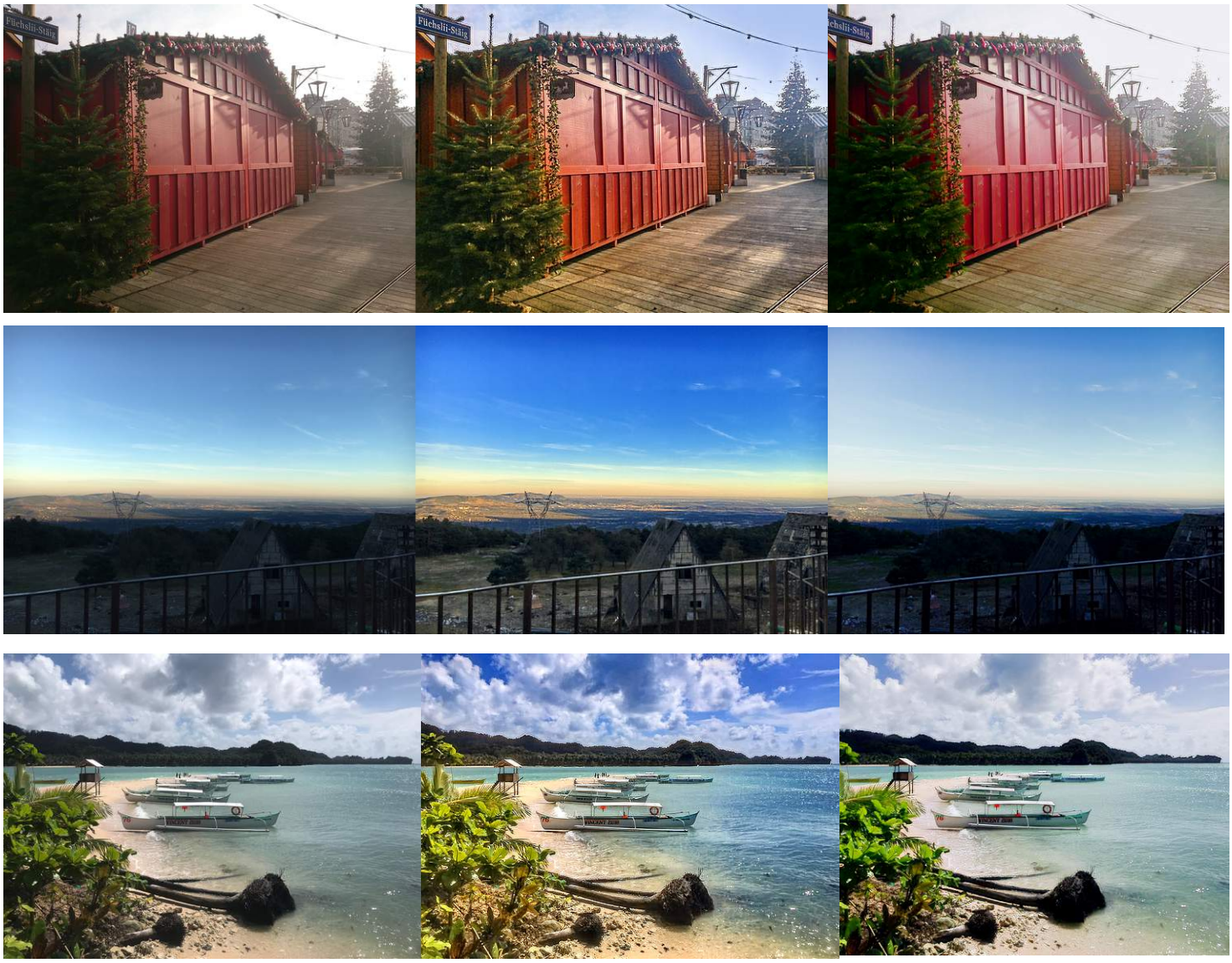

Fig 24. Photo enhancer app results using LPGAN_736 and LPGAN_577 frozen models. Left to right, top to bottom: Original RAMP test image / LPGAN_736 frozen model / LPGAN_577 frozen model

6. CONCLUSION AND FUTURE WORK

6.1 Conclusion

Deep Neural Networks have proved to be an excellent solution for photo enhancement thanks to the fast growing computational capacity. Traditional models such as MLP are not strong enough to process images with reasonable quality. Although there are supervised models like Convolutional Networks that have positioned themselves in state of the art when working with images, unsupervised models like Geneartive Adversarial Networks, are obtaining excellent results that are at the same level of the best convolutional architectures.

In particular, models based on GAN arquitectures, such as DCGAN or CycleGAN, are able not only to extract features in a group of images and recognise common patterns, but also to create completely new images with the extracted characteristics, on which you can see the same style of the images previously learnt. Moreover, when we are working with images, using GAN reduces the cost of recollection of an extended dataset, as unsupervised models do not require pairs of images during the training process.

The results obtained using the selected GAN models (LPGAN_736 y LPGAN_577) have been successful. In the generated images by LPGAN_736 and LPGAN_577, it can be appreciated the HDR style and professional editing style of MIT-5K dataset models have been trained with, respectively. Also, both models are the fastest to generate the photo enhancement, resulting on a perfect combination between efficiency and quality enhancement.

In terms of the mobile application development, the results of the frozen models, LPGAN_736 y LPGAN_577, has not been damaged in the transition to a mobile device. The enhancement in the mobile device is executed in a similar way and the results are even better appreciated in a mobile device than a computer screen. As expected, the execution time in a mobile device increases for both cases as the mobile capacity is yet far from the computer. However, LPGAN_736 model, obtained on average an execution time of 6 seconds, which is an excellent result given the phone limitations.

Therefore, the objectives set at the start of the project, have been successfully completed. However, the chosen models have limitations that are subjective to improve in future, these will be discussed on the following section.

6.2 Future work

The main limitation in the selected models for the photograph enhancement is the image resolution as they are set to work with photographs of 512x512. This means that input images need to be scaled down which is translated to an image quality loss. To solve this problem there are two possible solutions: i) Scale up image resolution once photograph has been enhanced. ii) To adapt the model so is set to work with photographs with unrestricted resolution.

For the first proposed solution, the existing super-resolution models, based on neural networks, could be applied into the already enhanced photograph. However, does not seem to be the most efficient solution, another model would need to be added into the existing one, which would increase the complexity of the solutions as well as the execution time on the mobile device. On the contrary, the second proposed solution seems more effective and elegant due to the development of the existing models able to process high-resolution image. However this would be a challenge for the mobile application.

On another hand, objective evaluation metrics are not reliable enough to evaluate improvements in photographs that are not related with noise or other artefacts that might influence in the quality. Although there are other traditional metrics such as polls or user tests, there are still no models that can generate and automatic evaluation based on a photography expert's opinion.

To conclude, an attempt to solve one of problems mentioned above took place in 2018 during PIRM Challenge [PIRM]. The competition was focused on image enhancement in mobile devices on the most efficient way, dividing the problems of super-resolution and photograph enhancement. For the evaluation of the solutions, it was used a combination of objective and subjective metrics that were able to measure the quality of enhanced image, the time spent and a subjective opinion of the users. The results obtained by the competition winners are extremely positive, since the best model in photo enhancement task performed the enhancement in just a few milliseconds. Also, as discussed in PIRM Challenge Competitions, mobile phones are more powerful everyday, which is eliminating limitation problems on mobile devices.

The numerous solutions to problems that used to be seen as impossible have been able to be executed thanks to the advances in Artificial Intelligence. Sometimes, it makes us reflect on what would the next steps be, leaving an uncertain but widely open future.

7. REFERENCES

[1] Photography School Spéos. Photo-museum. [Online]. http://www.photo-museum.org/niepce-invention-photography/#

[2] Kuldeep Narayan Shukla, Anjali Potnis, and Prashant Dwivedy, "A Review on Image Enhancement Techniques," *International Journal of Engineering and Applied Computer Science (IJEACS)*, vol. 02, no. 07, July 2017.

[3] Shikha Mahajan and Richa Dogra, "A Review on Image Enhancement Techniques," *International Journal of Engineering and Innovative Technology (IJEIT)*, vol. 4, no. 11, May 2015.

[4] Young Shik Moon, Bok Gyu Han, Hyeon Seok Yang, and Ho Gyeong Lee, "Low Contrast Image Enhancement Using Convolutional Neural Network with Simple Reflection Mode," *Advances in Science, Technology and Engineering Systems Journal* , vol. 4, no. 1, pp. 159-164, February 2019.

[5] Boletín Oficial del Estado Diciembre 2018. [Online]. https://www.boe.es/eli/es/lo/2018/12/05/3

[6] Github. Licensing a repository. [Online]. https://help.github.com/en/articles/licensing-a-repository#disclaimer

[7] AI Google. (2019) AI Applications. [Online]. https://ai.google/research/pubs/

[8] Marguerite Mcneal. (2015) Fei-Fei Li: If We Want Machines to Think, We Need to Teach Them to See. [Online]. https://www.wired.com/brandlab/2015/04/fei-fei-li-want-machines-think-need-teach-see/

[9] Standford University CA. (2019) Fei-Fei LI profile Standford University. [Online]. https://profiles.stanford.edu/fei-fei-li

[10] Rafael C. Gonzalez and Richard E. Woods, *Digital Image Processing*, 3rd ed., Marcia J. Horton et al., Eds. New Jersey, EEUU: Pearson Education, Inc, 2008.

[11] Shweta K.Narnaware and Roshni Khedgaonkar, "A Review on Image Enhancement Using Arificial Neural Network and Fuzzy Logic," *International Journal of Computer Science and Information Technologies (IJCSIT))*, vol. 6, no. 1, pp. 133-136, 2015.

[12] Ron Fedkiw and University CA, Standford. (2012) Image Processing and Related Fields. [Online]. https://web.stanford.edu/class/cs75n/Vision.pdf

[13] Adobe Photoshop Lightroom. [Online].

https://www.adobe.com/es/products/photoshop-lightroom.html

[14] Google. Google I/O recap. [Online]. https://events.google.com/io2018/recap/

[15] Warren S.McCulloch and Walter H.Pitts, "A Logical Calculus Of The Ideas Immanet in Nervous Activity," *Bulletin of Mathematical Biophysics*, vol. 5, pp. 115-133, 1943. [Online]. http://www.cse.chalmers.se/~coquand/AUTOMATA/mcp.pdf

[16] F Rosenblatt, "The Perceptron: a Probabilistic Model For Information Storage and Organization In The Brain," vol. 65, no. 6, pp. 386-407, 1958.

[17] Standford University CA. History: 1940's to the 1970's. [Online]. https://cs.stanford.edu/people/eroberts/courses/soco/projects/neural-networks/History/history1.html

[18] Alberto Quesada. 5 algorithms to train a neural network. [Online]. https://www.neuraldesigner.com/blog/5_algorithms_to_train_a_neural_network

[19] Syrine Ben Driss, Soua Mahmoud, Rostom Kachouri, and Mohamed Akil, "A comparison study between MLP and Convolutional Neural Network models for character recognition," *SPIE Conference on RealTime Image and Video Processing*, April 2017.

[20] Prabhu. Understanding of Convolutional Neural Network (CNN) — Deep Learning. [Online]. https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148

[21] Standford University Vision Lab. (2016) ImageNet. [Online]. http://image-net.org/

[22] ImageNet Classification with Deep Convolutional Neural Networks. (2012) [Online]. https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[23] Jie Hu, Li Shen, Samuel Albanie, Gang Sun, and Enhua Wu, "Squeeze-and-Excitation Networks," May 2019.

[24] Kevin P. Murphy, *Machine Learning: A Probabilistic Perspective*. Masachusets, United States, 2012.

[25] Sida Liu and Adrian Barbu, "Unsupervised Learning of Gaussian Mixture Models with a Uniform Background Component," *Journal of Machine Learning Research*, December 2018.

[26] Ian J. Goodfellow et.al, "Generative Adversarial Nets," June 2014.

[27] Ian Goodfellow, "NIPS 2016 Tutorial: Generative Adversarial Networks," April 2017.

[28] Branko Blagojevic. Generating Letters Using Generative Adversarial Networks. [Online]. https://medium.com/ml-everything/generating-letters-using-generative-adversarial-networks-gans-161b0be3c229

[29] Alec Radford, Luke Metz, and Soumith Chintala, "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks," *ICLR* , November 2015.

[30] Phillip Isola, Jun-Yan Zhu, Taesung Park, and Alexei A. Efros, "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks," 2018.

[31] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A. Efros, "Generative Visual Manipulation on the Natural Image Manifold," *European Conference on Computer Vision*, September 2016.

[32] Christian Ledig et al, "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network".

[33] Isola et al., "Image-to-Image Translation with Conditional Adversarial Networks," November 2016.

[34] Alain Horé and Djemel Ziou, "Image quality metrics: PSNR vs. SSIM," *Conference: 20th International Conference on Pattern Recognition*, 2010.

[35] Peter Ndajah et al., "An Investigation on The Quality of Denoised Images," *INTERNATIONAL JOURNAL OF CIRCUITS, SYSTEMS AND SIGNAL PROCESSING*, vol. 5, 2011.

[36] Zhou Wang, Alan Bovik, Hamid Rahim Sheikh, and Eero P. Simoncelli, "Image Quality Assessment: From Error Visibility to Structural Similarity," *IEEE Transactions on Image Processing*, May 2004.

[37] Kimhan Thung and Paramesran Raveendran, "A survey of image quality measures," *International Conference for Technical Postgraduates*, January 2010.

[38] Zhicheng Yan et al., "Automatic Photo Adjustment Using Deep Neural Networks," *ACM Transactions on Graphics*, May 2016.

[39] Michael Gharbi et al., "Deep Bilateral Learning for Real-Time Image

Enhancement," *ACM Transactions on Graphics*, July 2017.

[40] Vladimir Bychkovsky, Sylvain Paris, Eric Chan, and Fredo Durand. (2011) MIT-Adobe FiveK Dataset. [Online]. https://data.csail.mit.edu/graphics/fivek/

[41] Richard Zhang, Phillip Isola, and Alexei A. Efros, "Colorful Image Colorization," October 2016.

[42] Liang Shen et al., "MSR-net:Low-light Image Enhancement Using Deep Convolutional Network," *Huazhong University of Science and Technology*, November 2017.

[43] Kai Zhang et al., "Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising," August 2016.

[44] Zhigang Ling, Guoliang Fan, Yaonan Wang, and Xiao Lu, "Learning deep transmission network for single image dehazing," *IEEE International Conference on Image Processing*, September 2016.

[45] Ignatov et al., "DSLR-Quality Photos on Mobile Devices with Deep Convolutional Networks," 2017.

[46] Ignatov et al., "WESPE: Weakly Supervised Photo Enhancer for Digital Cameras," 2017.

[47] Agustsson, Eirikur, Timofte, and Radu. (2017, July) DIVerse 2K resolution high quality images as used for the challenges. [Online]. https://data.vision.ee.ethz.ch/cvl/DIV2K/

[48] Yu-Sheng Chen, Yu-Ching Wang, Man-Hsin Kao, and Yung-Yu Chuang, "Deep Photo Enhancer: Unpaired Learning for Image Enhancement from Photographs with GANs," 2018.

[49] Martin Arjovsky, Soumith Chintala, and Leon Bottou, "Wasserstein GAN," December 2017.

[50] Olaf Ronneberger, Philipp Fischer, and Thomas Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," May 2015.

[51] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep Residual Learning for Image Recognition," December 2015.

[52] Tensorflow Library. [Online]. https://www.tensorflow.org/

[53] DSLR-Quality Photos on Mobile Devices with Deep Convolutional Networks

repository. [Online]. https://github.com/aiff22/DPED

[54] Deep Photo Enhancer: Unpaired Learning for Image Enhancement from Photographs with GANs repository. [Online]. https://github.com/nothinglo/Deep-Photo-Enhancer

[55] Gadgets Now. [Online]. https://www.gadgetsnow.com/laptops/Lenovo-Thinkpad-X270-20HMA06XIG-Laptop-Core-i5-7th-Gen8-GB1-TBWindows-10

[56] PSNR implementation. [Online]. https://github.com/aizvorski/video-quality/blob/master/psnr.py

[57] Sckit-image image processing python. [Online]. https://scikit-image.org/

[58] SSIM implentation Sckit-image. [Online]. https://github.com/scikit-image/scikit-image/blob/master/skimage/measure/_structural_similarity.py

[59] Snapdragon Neural Processing Engine SDK. [Online]. https://developer.qualcomm.com/docs/snpe/index.html

[60] Snapdragon Neural Processing Engine SDK Reference Guide Supported Network Layers. [Online]. https://developer.qualcomm.com/docs/snpe/network_layers.html

[61] How to Use TensorFlow Mobile in Android Apps. [Online]. https://code.tutsplus.com/tutorials/how-to-use-tensorflow-mobile-in-android-apps--cms-30957

[62] Salarios para empleos de Programador/a junior en España-Indeed. [Online]. https://www.indeed.es/salaries/Programador/a-junior-Salaries

[63] Salarios para empleos de Jefe de proyecto en España-Indeed. [Online]. https://www.indeed.es/salaries/Jefe-de-proyecto-Salaries

[64] Harmandeep Kaur Ranota and Prabhpreet Kaur, "Review and Analysis of Image Enhancement Techniques," *International Journal of Information & Computation Technology*, vol. 4, no. 6, pp. 583-590.

[65] Avneet Pannu and M. Tech Student , "Artificial Intelligence and its Application in Different Areas," *International Journal of Engineering and Innovative Technology (IJEIT)*, vol. 4, no. 10, April 2015.

[66] Dr. Chris Y. H. Chan and Hong Kong Polytechnic University. Image Enhancement. [Online]. http://www.eie.polyu.edu.hk/~enyhchan/imagee.pdf

[67] Ricardo Cancho Niemietz. (2008, March) Wikimedia Commons. [Online]. https://commons.wikimedia.org/wiki/File:RGB_channels_separation.png

[68] Himanshu Singh and Prof.Vivek Singh, "A Comparative Analysis on Histogram Equalization Techniques for Medical Image Enhancement ," *International Journals of Advanced Research in Computer Science and Software Engineering*, vol. 7, no. 6, June 2017.

[69] Nasim Mansurov. Photography Life. [Online]. https://photographylife.com/lightroom-before-and-after

[70] Awhan Mohanty. Becoming Human AI. [Online]. https://becominghuman.ai/multi-layer-perceptron-mlp-models-on-real-world-banking-data-f6dd3d7e998f

[71] Muhammad Moinuddin, Ubaid M Al-Saggaf, and Syed Saad Azhar Ali, "Contrasting Convolutional Neural Network (CNN) with Multi-Layer Perceptron (MLP) for Big Data Analysis," *International Conference on Intelligent and Advanced System (ICIAS)*, 2018.

PROJECT BUDGET

The project budget presented below includes an estimation of the time and material costs that a company may incur in case of developing the proposed study. The proposed budget includes three different roles:

- Research Manager: head of the research which is in charge of the planification and main research about the topic of the project. The Research Manager has to be highly experienced in machine-learning theory and modelling, with team management abilities to check regulary the progress of the project, and verify the solution developed by the programmers.
- AI Programmer: software developer with knowledge in Aritificial Intelligence and technical skills to develope AI models, and highly familiar with libraries as Tensorflow, Keras, Pytorch… .
- Software Programmer: software developer which purpose is to give support to the AI programmer with the pure code implementation of the model. The programmer has to be also well experienced with Android application development.

TABLE XVI

ESTIMATED HUMAN CAPITAL COST

Position	**Hourly rate**	**Hours**	**Total**
Research Manager	19,5 €/hour	320 (8 weeks)	6.240 €
AI Programmer	12,6 €/hour	240 (6 weeks)	3.024 €
Software Programmer	10,5 €/hour	240 (6 weeks)	2.520 €
			11.784 €

The original project has been studied and developed by one person, in a period of 8 monthsa and working on a part-time basis. So, it is estimated that a three people team working full-time, where each member is focused on an specific task of the project, may reduce a third the project duration (2 months).

The workday for all the team memebers is 8 hours and a total of 40 hours per week. The hourly rate of programmers and head of research presented in Table XVI is estimated from [**62**] [**63**] as the mean annual wage these positions receive in Spain. Therefore, the total cost of human capital is estimated in 11.784 €.

TABLE XVII

ESTIMATED MATERIAL COST

Description	**Purchase price**	**Amortization**	**Usage**	**Amount**	**Total cost**
Laptop (Lenovo ThinkPad X270)	899,99 €/	15 €/month	2 months	3	90 €
Windows 10 Home	0,0 € (included in laptop price)	-	2 months	3	0 €
Tensorflow	0,0 €	-	2 months	3	0 €
Github	0,0 €	-	2 months	3	0 €
					90 €
EXTRA					
Google Cloud	20 €/mes	-	2 months	1	40 €
Google Drive	1,99 €/mes	-	2 months	3	5,97 €
					45,97 €

On the other hand, Table XVII shows a list of software and hardware materials necessary in the development of the project. The requiered materials have been selected according to the ones used during the development of this thesis. Hardware costs are valued as the product of the amortization (assuming 5 years of shelf life) and the months of usage. There are extra materials that were not used in this thesis that might be helpful for others in terms of logistic and efficience when working with code and AI models.

TABLE XVIII

PROJECT BUDGET SUMMARY AND TOTAL COST

Description	**Cost**
Human capital	11.784 €
Materials	90 €
Extra materials	45,97 €
Total cost before taxes	**11.919,97 €**
Taxes (21%)	2.503,19 €
Total cost after taxes	**14.423,16 €**

The total cost of the project before taxes, summing human and material costs, is shown in the table below. Taxes are applied according to the Spanish tax IVA (21%). Finally, the cost after taxes and including extra materials is **14.423,16 €**.

PROJECT PLAN

In order to make the project plan, first the necessary tasks are specified:

1. Analysis and resarch of the topic
2. Requisites document elaboration for programmers
3. Development of the model
4. Supervision of the project
5. Evaluation plan
6. Validation of the model and feedback
7. Model amendments
8. Final submission and last revision

Next, a Gantt chart is made to better visualize these activities in the course of the project. The three roles needed in the project are represented using different colors. Blue for Research Manager, yellow for both AI Programmer and Software Programmer, and green for joint tasks by the three members of the project.

Task name	**Month 1**				**Month 2**			
	1	**2**	**3**	**4**	**1**	**2**	**3**	**4**
Analysis and research of the topic								
Requisites document elaboration for programmers								
Evaluation plan elaboration								
Development of the model								
Supervision of the project								
Validation of the model and feedback								
Model amendments								
Final submission and last revision								

APPENDIX

Visual results DPED test frozen LPGAN_736 on BQ Aquaris V.

Visual results DPED test frozen LPGAN_577 on BQ Aquaris V.


Fuchslii-Stäig
Fuchslii-Stäig

Visual results RAMP test frozen LPGAN_736 on BQ Aquaris V.

Visual results RAMP test frozen LPGAN_577 on BQ Aquaris V.